# A Deniably Authenticated Searchable Public Key Encryption Scheme in Mobile Electronic Mail System

Shuhan Zeng, Yongjian Liao *Member, IEEE*, Chuanhao Zhou, Jinlin He, and Hongwei Wang

***Abstract*—Confidentiality and authentication are two main security goals in secure electronic mail (e-mail). Furthermore, deniability is also a significant security property for some e-mail applications to protect the privacy of the sender. Although searchable encryption solves the keyword searching problem in a secure e-mail system, it also breaks the deniability of the system. Because the adversary can obtain the information of the data sender and data user from the trapdoor as well as ciphertext used for keyword searching. At the same time, efficiency is another problem in mobile computing. To overcome the issues, we put forward deniably authenticated searchable public key encryption (DA-SPKE) which can apply to the deniable e-mail system. We first define the DA-SPKE model and propose the DA-SPKE scheme. Then we prove its security in the random oracle model and evaluate its efficiency. Finally, we use it in a deniably secure e-mail system to protect both data senders' and data users' privacy.**

***Index Terms*—Searchable public key encryption, Deniability, Authentication, E-mail, Privacy, Keyword guessing attack, Chosen keyword attack.**

## I. Introduction

ELECTRONIC Mail (E-mail) is a method of electronic information exchange transmitted through computer networks. More and more important information can be exchanged by e-mail. People send and read e-mails from mobile phones, laptops, and personal computers. The great convenience for exchanging information also brings a lot of challenges to security and efficiency. Secure e-mail protocols (or systems), such as Pretty Good Privacy (PGP) [1] and Secure/Multipurpose Internet Mail Extensions (S/MIME) [2], focus on the confidentiality and authentication of the e-mail contents. However, these protocols are inevitable to leak the identity of the sender for the signature scheme is used to sign the content of e-mails in them. To solve the issue, secure deniably authenticated encryption schemes (DAE) [3]–[7] were proposed and secure-deniable e-mail systems were setup recently [2], [8], [9].

A secure-deniable e-mail system addresses the security of e-mail content, such as confidentiality, authentication, and deniability. But a new challenge arises – How to search some words (keywords) without revealing the privacy of a mass of e-mail letters in a secure-deniable e-mail system when the user wants to find out some e-mail letters? Searchable symmetric key encryption (SSKE) [10] and searchable public key encryption (SPE) for emails [11]–[13] can partially address this issue. Since the ciphertexts of context append the ciphertexts of keywords stored in the e-mail server (or cloud servers), the e-mail server can do some operations on the ciphertexts of keywords without obtaining any information of keywords to retrieve the corresponding e-mails. However, the existing schemes and related schemes cannot protect the privacy of the sender, which would reveal the identity of the sender from ciphertexts of the keywords or the trapdoors of keywords. This leads to the revelation of sender and receiver and the deniability of the secure-deniable e-mail system is broken. Thus, the existing solutions are not fit for the secure-deniable e-mail system. With the development of modern science and technology, there is a growing preference for portable devices, such as mobile phones, tablet personal computers, and laptops, to handle email. Efficiency is another requirement of some schemes. We attempt to solve the above issues.

As shown in Fig. 1, users of email system use email apps on their phones, laptops, and pads to send, receive, and retrieve emails through the outsourced cloud in a secure-deniable email system. Retrieving emails in the cloud on mobile devices is always implemented through authenticated SPKE schemes. But in the classic authenticated SPKE scheme, the role of the user is always known by others. The role of the data sender is always to encrypt and upload keywords as well as generate trapdoors, and the role of the data user upload the trapdoor. It's an information leakage of system users to the third party in a sense. From this perspective of view, we come up with the following security properties for a secure-deniable e-mail system in mobile computing:

- Confidentiality. Encrypted keywords should not leak any information to illegal and unauthorized users.
- Authentication. Ensure that data originates from authorized users and prevent malicious users from transmitting data.
- Security against keyword guessing attack. It indicates that a malicious attacker is not able to guess some candidate keywords and verify whether his guess is correct or not.
- Deniability. The data sender and data user can be both deniable to the ciphertexts or trapdoors that they generate. That is, both the data sender and data user can produce valid ciphertexts and trapdoors for any keywords.

The first three are fundamental security properties, and the system's deniability is special and unique. Confidentiality

This paper was produced by the Cryptography Group in Space Security Lab in School of Information and Software Engineering at University of Electronic Science and Technology of China (UESTC). They are in Chengdu City, China.

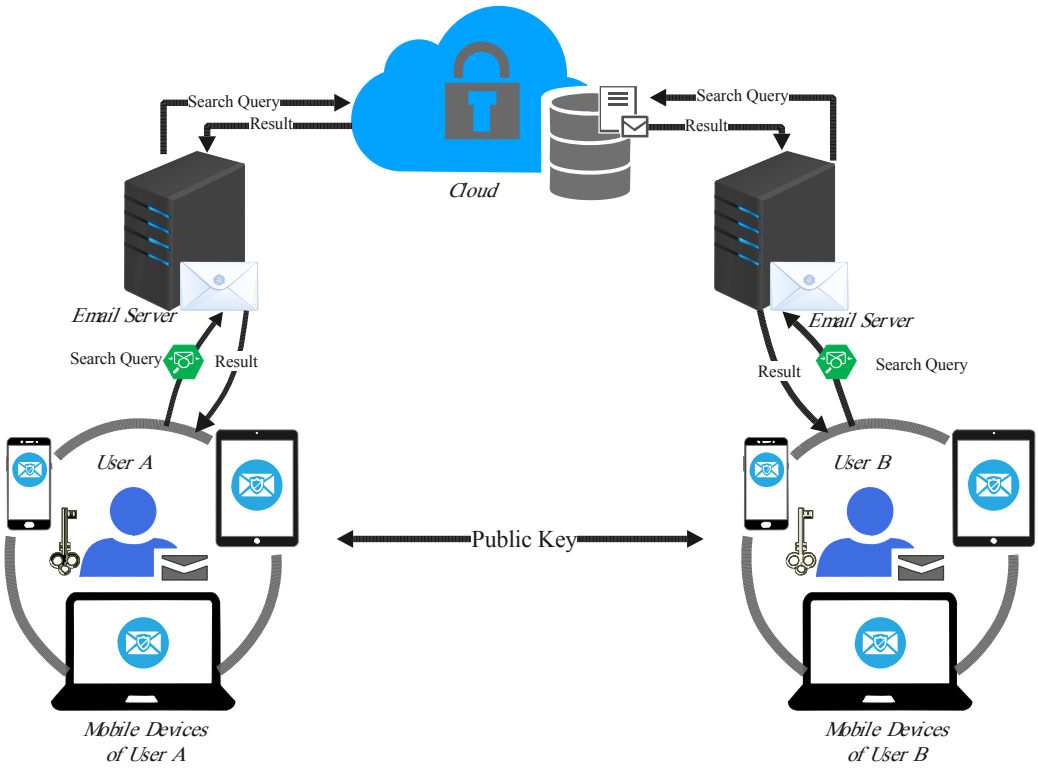



Fig. 1. System model

should be achieved by indistinguishability against chosen keyword attack game (IND-CKA). Security against KGA is achieved by indistinguishability against keyword guessing attack (IND-KGA). Specifically, IND-CKA infers that the encrypted keywords leak no information about the keywords themselves, and IND-KGA infers that the trapdoors corresponding to keywords leak no information about the keywords themselves. KGA in SPKE is of great importance in many same types of works [14]–[17]. Authentication [18]–[21] is implicitly achieved in our scheme, where only the data user and data sender can generate valid ciphertexts and trapdoors to test the consistency of keywords behind the scene. Deniability is a natural property in the generation of ciphertexts and trapdoors by using Diffie-Hellman shared secret key. The concept of deniability always appears in public key authenticated deniable encryption (PKADE) [22]–[24], where deniability only needs to affect ciphertext because there is no trapdoor generation algorithm in PKADE.

### A. Contributions

In this paper, we propose a deniable authenticated searchable public key encryption (DA-SPKE) applied to a secure-deniable email system for mobile computing. To the best of our knowledge, this is the first to take into account the deniability of searchable encryption. Our main contributions can be described below.

- Firstly, we put forward the primitive of DA-SPKE, which is similar to DAE except that it is required to satisfy the deniability of keywords and trapdoors, and then we define the security model, such as IND-CKA and IND-KGA.
- Then, we constructed a DA-SPKE scheme from bilinear pairing and proved that the DA-SPKE scheme is secure for IND-CKA and IND-KGA in the random oracle. IND-CKA indicates that ciphertexts of the scheme leak no information about the corresponding keywords. IND-KGA means that the attacker gains no information from the trapdoors.
- Next, we have the comparison of performance among related schemes [25]–[31] and it showed that our proposed scheme still gains an advantage, although deniability of the scheme is required.
- Finally, we designed a secure email searchable keyword protocol based on our encrypted keywords and trapdoors in the secure and deniable e-mail system.

### B. Related works

In this section, we focus on existing SPKE schemes, authenticated SPKE schemes, and DA-SPKE schemes. In 2000, Song et al. [10] proposed a pioneering work in searchable encryption which is the searchable symmetric encryption (SSE). But symmetric encryption means the management and distribution of secret keys is inevitable, which costs a lot. Therefore, people started to study searchable public key encryption (SPKE). In 2004, Boneh et al. [25] proposed a scheme based on the concept of public key encryption with keyword search (PEKS). But most of the SPKE schemes proposed then were not able to resist KGA and CKA by insider adversaries [32]. Many great works [26]–[31] proposed schemes that are against KGA, where curious and unbelievable but honest cloud find no information of keywords by KGA. By the way, SPKE is also widely used in other applications such as wireless sensor network [33], blockchain-based vehicular social networks [34], industrial internet of things [35] and mobile health care system [36].

To enable authorized rather than unauthorized users to conduct searches on Specific keywords, the authenticated SPKE scheme has emerged prominently in the research on searchable encryption. Liu [3] comes up with an authenticated searchable encryption scheme with aggregate key, which accomplishes multiple users in the system. Mihailescu and Nita [4] combine two authentication means in their SPKE scheme to improve security. Chen et al. [5] put forward an idea of password-authenticated searchable encryption where only the legitimate user who knows the initially registered password can perform operations of the scheme, which originated from password-based protocols for authentication and secret sharing. Some works focus on SPKE schemes in IoT applications [6], [7]. Lu and Li [37] devised a lightweight authenticated SPKE against adaptively-chosen-targets adversaries for mobile devices. Miao et al. [38] design a revocably time-controllable keyword search scheme in mobile e-health cloud. Liu et al. [39] proposed a dynamic searchable symmetric encryption scheme with forward and backward privacy in secure data outsourcing. Zhang et al. [40] brought up a multi-user certificatelessly deniably authenticated searchable encryption scheme, where their key operations need the participation of the third party. Still, the concept of their deniability only concentrates on encryption by using hash function, the trapdoor generation is not covered in deniability, which makes the scheme not fully deniable.

### C. Organization

The remainder of the contents are structured below. We will recall some basic concepts in section 2. After that, we introduce the system model and security model in section 3. Our concrete DA-SPKE scheme and its secure and performance analysis are provided in section 4. Finally, in section 5, we conclude this paper.

## II. Preliminaries

In this section, we will introduce some basic concepts used in this paper.

### A. Notations

Table I describes the notations used throughout the paper.

TABLE I
NOTATIONS AND DESCRIPTIONS

| Notations | Descriptions |
|---|---|
| $k$ | security parameter |
| $\mathbb{G}_1, \mathbb{G}_T$ | cyclic groups |
| $e$ | bilinear map |
| $H, H_1$ | hash functions |
| $r \xleftarrow{R} \mathbb{Z}_p^*$ | randomly choose $r$ from $\mathbb{Z}_p^*$ |
| [1,n] | {1,2,...,n} |
| DS | data sender |
| DU | data user |
| CSP | cloud service provider |
| DH-SSK | Diffie-Hellman Shared Secret Key |
| $pk_i$ | user i's public key |
| $sk_i$ | user i's secret key |
| $w_i$ | $i^{th}$ keyword in a keyword list |
| T | trapdoor |
| C | ciphertext |

### B. Bilinear Map

Let $\mathbb{G}_1$ be an additive cyclic group and $\mathbb{G}_T$ be a multiplicative cyclic group which satisfy the bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_T$, with prime order $p$. Let $P$ be a random generator of $\mathbb{G}_1$. The bilinear map $e$ has the following properties:

- **Bilinear**: The equation $e(aP, bP) = e(P, P)^{ab}$ holds for any $a, b \in \mathbb{Z}_p^*$ and $P \in \mathbb{G}_1$.
- **Non-degeneration**: If $P \neq 1_{\mathbb{G}_1}$, we have $e(P, P) \neq 1_{\mathbb{G}_T}$, where $1_{\mathbb{G}_1}$ and $1_{\mathbb{G}_T}$ respectively represent identity element of $\mathbb{G}_1$ and $\mathbb{G}_T$.
- **Computability**: $e$ is efficiently computable.

### C. Diffie-Hellman Shared Secret Key

*Definition 1 (Diffie-Hellman Shared Secret Key):* Let Alice and Bob as users respectively have the public/private key pairs ($aP$, $a$), ($bP$, $b$), where $P \in \mathbb{G}_1$ is a generator of the cyclic group $\mathbb{G}_1$ with order $p$ and $a, b \in \mathbb{Z}_p^*$ are chosen by Alice and Bob respectively and secretly. It follows that Alice and Bob have a shared secret key (DH-SSK) $K = abP$, which derives from the key agreement protocol proposed by Diffie and Hellman [41].

### D. Computational Diffie-Hellman assumption (CDH)

Let $\mathbb{G}_1$ be a cyclic group with prime order $p$, a generator of which is $P$. Given a tuple $(P, aP, bP)$, where $a, b$ are randomly chosen from $\mathbb{Z}_p^*$. Here we define the advantage of solving CDH problem for any probabilistic polynomial time (PPT) adversary $\mathcal{A}$.

$$Adv_{\mathcal{A}}^{CDH}(k) = Pr[\mathcal{A}(P, aP, bP) \rightarrow abP]$$

The CDH assumption is that there is no PPT algorithm to get $abP$ in a non-negligible advantage. i.e., CDH assumption holds if $Adv_{\mathcal{A}}^{CDH}(k)$ is negligible in the security parameter $k$.

### E. System Model

Our system model consists of the following seven algorithms.

- **Setup**$(1^k)$: After taking as input the security parameter $k$, it generates the public parameter $Para$.
- **KeyGen**$(Para)$: The DS and the DU take as input the system parameter $Para$ then generates public/secret key pairs $(pk_s, sk_s)$, $(pk_u, sk_u)$ respectively. Then DU and DS send their public keys to the CSP.
- **DA-SPKE**$(Para, pk_u, sk_s, w_i)$: It takes as input the system parameter $Para$, DU's public key $pk_u$, DS's secret key $sk_s$ and the keyword $w_i$, then generates the ciphertext corresponding to $w_i$.
- **Trapdoor**$(Para, pk_s, sk_u, w_j)$: It takes as input the system parameter $Para$, DS's public key $pk_s$, DU's secret key $sk_u$ and the keyword $w_j$, then generates a trapdoor $T_j$ corresponding to $w_j$.
- **Test**$(Para, T, C)$:It takes as input the system parameter $Para$, a trapdoor $T$ and a ciphertext $C$, then outputs 1 if the contents behind $C$ and $T$ are the same; Otherwise outputs 0.
- **Deniable DA-SPKE**$(Para, pk_s, sk_u, w_i)$: It takes as input the system parameter $Para$, DS's public key $pk_s$, DU's secret key $sk_u$ and the keyword $w_i$, then generates the ciphertext corresponding to $w_i$.
- **Deniable Trapdoor**$(Para, pk_u, sk_s, w_j)$: It takes as input the system parameter $Para$, DU's public key $pk_u$, DS's secret key $sk_s$ and the keyword $w_j$, then generates a trapdoor $T_j$ corresponding to $w_j$.

### F. Security Model

In this subsection, we will introduce two security models: IND-CKA security model and IND-KGA security model. To illustrate, the IND-CKA security model signifies that any PPT adversary could obtain no information from the ciphertext of the keyword and the IND-KGA security model indicates that any PPT adversary could not obtain any information from the trapdoor of the keyword.

*1) IND-CKA Security Model:* This is a game between challenger $\mathcal{B}$ and the adversary $\mathcal{A}$. The aim of $\mathcal{A}$ is to distinguish ciphertexts of two keywords and $\mathcal{B}$ establishes the system and answers queries asked by $\mathcal{A}$.

- **Setup**: The challenger $\mathcal{B}$ selects a security parameter $k$ and generates the public parameter $Para$ by $Setup$ algorithm. Then $\mathcal{B}$ performs $KeyGen$ algorithm to generate the public-secret key pairs $(pk_s, sk_s)$ and $(pk_u, sk_u)$ respectively for data sender (DS) and data user (DU). Finally the challenger $\mathcal{B}$ makes $Para$ and public keys which is $pk_s$ and $pk_u$ known to the public.
- **Phase I**: The challenger $\mathcal{B}$ answers $\mathcal{A}$'s adaptive queries below in polynomial time:
  - Ciphertext query: The adversary $\mathcal{A}$ gives the challenger $\mathcal{B}$ a keyword $w$. Then $\mathcal{B}$ executes the encryption algorithm (either the original one or deniable one) and returns the output $C_w$ which is the ciphertext of $w$ back to the adversary $\mathcal{A}$.
  - Trapdoor query: The adversary $\mathcal{A}$ gives the challenger $\mathcal{B}$ a keyword $w$. Then $\mathcal{B}$ executes the algorithm (either the original one or deniable one) and returns the output $T_w$ which is the trapdoor of $w$ back to the adversary $\mathcal{A}$.
- **Challenge**: $\mathcal{A}$ chooses two keywords $w_0, w_1$ that have not been queried for a trapdoor and sends them to the challenger $\mathcal{B}$. Then $\mathcal{B}$ randomly chooses $b \in \{0, 1\}$ then generates the challenge ciphertext $C_b$ and sends it to $\mathcal{A}$.
- **Phase II**: This stage is the same as Phase $I$. While the only restriction is that neither $w_0$ nor $w_1$ should be queried to the trapdoor query.
- **Guess**: $\mathcal{A}$ outputs her guess $b' \in \{0, 1\}$ for $b$ and wins the game if $b' = b$.

The advantage for $\mathcal{A}$ to win the game is defined as:

$$Adv_{\mathcal{A}}^{IND\text{-}CKA}(k) = |Pr[b = b'] - \frac{1}{2}|.$$

*Definition 2:* A searchable public key encryption scheme is secure under indistinguishability against chosen keyword attack if the advantage of the adversary is negligible in the above game.

*2) IND-KGA Security Model:* This is a game between challenger $\mathcal{B}$ and the adversary $\mathcal{A}$ referring to [42], and Wu et al.'s scheme also needs to satisfy it. The aim of $\mathcal{A}$ is to distinguish trapdoors of two keywords and $\mathcal{B}$ establishes the system and answers queries asked by $\mathcal{A}$.

- **Setup**:The challenger $\mathcal{B}$ selects a security parameter $k$ and generates the public parameter $Para$ by $Setup$ algorithm. Then $\mathcal{B}$ performs $KeyGen$ algorithm to generate the public-secret key pairs $(pk_s, sk_s)$ and $(pk_u, sk_u)$ respectively for DS and DU. Finally the challenger $\mathcal{B}$ makes $Para$ and public keys which are $pk_s$ and $pk_u$ known to the public.
- **Phase I**: The challenger $\mathcal{B}$ answers $\mathcal{A}$'s adaptive queries below in polynomial time:
  - Ciphertext query: The adversary $\mathcal{A}$ gives the challenger $\mathcal{B}$ a keyword $w$. Then $\mathcal{B}$ executes the encryption algorithm (either the original one or deniable one) and returns the output $C_w$ which is the ciphertext of $w$ back to the adversary $\mathcal{A}$.
  - Trapdoor query: The adversary $\mathcal{A}$ gives the challenger $\mathcal{B}$ a keyword $w$. Then $\mathcal{B}$ executes the algorithm (either the original one or deniable one) and returns the output $T_w$ which is the trapdoor of $w$ back to the adversary $\mathcal{A}$.
- **Challenge**: $\mathcal{A}$ chooses two keywords $w_0, w_1$ that have not been queried for ciphertext and sends them to the challenger $\mathcal{B}$. Then $\mathcal{B}$ randomly chooses $b \in \{0, 1\}$ then generates the challenge trapdoor $T_{w_b}$ and sends it to $\mathcal{A}$.
- **Phase II**: This stage is the same as Phase I. While the only restriction is that neither $w_0$ nor $w_1$ should be queried to the ciphertext query and trapdoor query.
- **Guess**: $\mathcal{A}$ outputs her guess $b' \in \{0, 1\}$ for $b$ and wins the game if $b' = b$.

The advantage for $\mathcal{A}$ to win the game is defined as:

$$Adv_{\mathcal{A}}^{IND\text{-}KGA}(k) = |Pr[b = b'] - \frac{1}{2}|.$$

*Definition 3:* A searchable public key encryption scheme is secure under IND-KGA if the advantage of the adversary is negligible in the above game.

## III. Concrete deniable searchable public key encryption scheme

Our scheme consists of the seven PPT algorithms below.

### A. Construction

- **Setup**$(1^k)$: The algorithm takes as input a security parameter $k$ . It initially generates two cyclic groups $\mathbb{G}_1, \mathbb{G}_T$ with the same prime order $p$ ($|p| = k$), where $\mathbb{G}_1$ and $\mathbb{G}_T$ satisfy the bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_T$. Later it picks a generator $P$ of $\mathbb{G}$ as well as two hash functions $H : \mathbb{G}_1 \to \mathbb{Z}_p^*$ and $H_1 : \{0,1\}^* \to \mathbb{Z}_p^*$. Finally, the algorithm outputs the public system parameter $Para = (\mathbb{G}_1, \mathbb{G}_T, p, P, e, H, H_1)$.
- **KeyGen**$(Para)$: DU and DS execute $KeyGen$ to generate the public-secrete key pairs respectively. DS's public key $pk_s = aP$ and DU's public key $pk_u = bP$ are corresponding to the DS's secret key $sk_s = a$ and DU's secret key $sk_u = b$ respectively, where $a, b \in \mathbb{Z}_p^*$ are chosen randomly. At the end of the algorithm, the two roles of the system send the public keys $pk_s = aP$ and $pk_u = bP$ to the CSP of the system.
- **DA-SPKE**$(Para, pk_u, sk_s, w_i)$: DS runs this algorithm to generate the ciphertext of the given keyword $w_i$. DS computes the ciphertext by the given inputs, the algorithm works as follows:
  - DS selects a random number $r_i \in \mathbb{Z}_p^*$ for every keyword $w_i$ where $i \in [1, n]$.
  - DS computes $C_{1w_i} = r_i P$ and $C_{2w_i} = r_i sk_s pk_u = r_i abP$.
  - DS computes $C_{3w_i} = e(pk_u, pk_s)^{r_i H_1(w_i)^{H(sk_s pk_u)}}$.
  - DS assembles the ciphertext $C_{w_i} = (C_{1w_i}, C_{2w_i}, C_{3w_i})$ of the keyword $w_i$.

  Finally, DS uploads the ciphertext organized in some way to the cloud. It is worth mentioning that the DH-SSK $abP$ can be precomputed to lower the computational overhead and improve efficiency.
- **Trapdoor**$(Para, pk_s, sk_u, w_j)$: This algorithm is executed by the DU and works as follows:
  - DU selects a random number $r'_j \xleftarrow{R} \mathbb{Z}_p^*$ for some given keyword $w_j$ where $j \in [1, n]$.
  - DU computes $T_{1w_j} = r'_j P$ and $T_{2w_j} = (r'_j + H_1(w_j)^{H(sk_u pk_s)}) sk_u pk_s$.
  - DU assembles the trapdoor $T_{w_j} = (T_{1w_j}, T_{2w_j})$ corresponding to $w_j$.

  Finally, DU uploads the trapdoor to the CSP. It is worth mentioning that the DH-SSK $abP$ can be precomputed to lower the computational overhead and improve efficiency.
- **Test**$(Para, T_{w_j}, C_{w_i})$: This algorithm is executed by the CSP. The algorithm works as follows:
  - Compute $E_1 = e(T_{1w_j}, C_{2w_i})$;
  - Compute $E_2 = e(C_{1w_i}, T_{2w_j})$;
  - Compute $E_3 = \frac{E_2}{E_1}$;
  - Check if $E_3 = C_{3w_i}$ holds. If the equation holds, output 1 and send the files corresponding to the keyword to DU; Otherwise output 0 and terminate.

The deniability of the encryption algorithm and trapdoor generation algorithm are shown below. It indicates that the data user can generate completely legal ciphertexts by deniable DA-SPKE, and the data sender can also generate legal trapdoors by deniable Trapdoor. Both DU and DS can generate legal trapdoors and ciphertexts, which accomplishes the deniability of the scheme. The property of deniability is naturally achieved by utilization of Diffie-Hellman shared secret key. The two algorithms of deniability are listed below.

- **Deniable DA-SPKE**$(Para, pk_s, sk_u, w_i)$: The encryption algorithm with the property of deniability can also be executed by the DU. The algorithm works as follows, which is almost the same as **DA-SPKE** algorithm with only a slight difference in that the public and private keys as the input are distinctive:
  - DU selects a random number $r_i \in \mathbb{Z}_p^*$ for every keyword $w_i$ where $i \in [1, n]$.
  - DU computes $C_{1w_i} = r_i P$ and
$$C_{2w_i} = r_i sk_u pk_s = r_i abP$$
.
  - DU computes $C_{3w_i} = e(pk_u, pk_s)^{r_i H_1(w_i)^{H(sk_u pk_s)}}$.
  - DS sets $C_{w_i} = (C_{1w_i}, C_{2w_i}, C_{3w_i})$ as the ciphertext of the keyword $w_i$.

  Finally, DU uploads the ciphertext organized in some way to the cloud.
- **Deniable Trapdoor**$(Para, pk_u, sk_s, w_j)$: The trapdoor generation algorithm with the property of deniability can also be executed by the DS. The algorithm works as follows, which is almost the same as **Trapdoor** algorithm with only a slight difference that the public and private key as the input are distinctive:
  - DS selects a random number $r'_j \in \mathbb{Z}_p^*$ for some given keyword $w_j$ where $j \in [1, n]$.
  - DS computes $T_{1w_j} = r'_j P$ and $T_{2w_j} = (r'_j + H_1(w_j)^{H(sk_s pk_u)}) sk_s pk_u$.
  - DS assembles the trapdoor $T_{w_j} = (T_{1w_j}, T_{2w_j})$ corresponding to $w_j$.

  Finally, DU uploads the trapdoor to the CSP.

### B. Deniability

Imagine a scene where there are only two roles (i.e. DS and DU) in a communication, and at the same time, there is a party called the outsider watching these two roles. The two roles know who is the sender or receiver of the message, but what if they want the outsider know nothing about their roles? That is, from the outsider's perspective, he knows nothing about the state of the two roles, and whether one of the roles is the data sender is a Schrodinger's cat for him. In other words, the outsider has no idea who exactly masters the knowledge of the message they transmitted.

Our scheme makes this scene come true. The symmetry of the DH-SSK is the kernel of our asymmetric scheme. Based

on this characteristic, both DS and DU have a shared secret key, therefore they are both capable of generating ciphertexts and trapdoors for all keywords as well as making our scheme work. The outsider can no longer decide who exactly knows the transmitted message. From the perspective of DU and DS, they can deny themselves as DU or DS, and this property is called deniability.

### C. Correctness

In this section, we will supply the correctness analysis of our scheme i.e. prove only when the keyword $w_j$ chosen by the data user is consistent with $w_i$ of the ciphertext will the cloud server send back the corresponding data to the data user.

$$\begin{aligned}
E_1 =& e(T_{1w_j}, C_{2w_i}) = e(r'_j P, r_i abP) \\
E_2 =& e(C_{1w_i}, T_{2w_j}) = e(T_{2w_j}, C_{1w_i}) \\
=& e(r_i P, r_j abP + H_1(w_j)^{H(abP)} abP) \\
E_3 =& \frac{E_2}{E_1} = \frac{e(r_i P, r_j abP + H_1(w_j)^{H(abP)} abP)}{e(r'_j P, r_i abP)} \\
=& \frac{e(r_i P, r'_j abP) e(r_i P, H_1(w_j)^{H(abP)} abP)}{e(r'_j P, r_i abP)} \\
=& e(r_i P, H_1(w_j)^{H(abP)} abP) \\
C_{3w_i} =& e(pk_u, pk_s)^{r_i H_1(w_i)^{H(pk_u sk_s)}} \\
=& e(r_i P, H_1(w_i)^{H(abP)} abP)
\end{aligned}$$

If and only if $w_i = w_j$, the equation $E_3 = C_{3w_i}$ will hold. Consequently, our scheme satisfies the correctness.

### D. Security Analysis

We now prove that our improved scheme is IND-CKA, IND-KGA secure deniability secure.

*Theorem 1:* Our improved scheme is IND-CKA security if the CDH assumption holds. In other words, for any PPT adversary $\mathcal{A}$, her advantage $Adv_{\mathcal{A}}^{IND\text{-}CKA}$ in IND-CKA security game is negligible.

**Proof**:We define four security games – **Game 0**, **Game 1**, **Game 2** and **Game 3** below.

**Game 0**.

- $Setup$. The challenger $\mathcal{B}$ selects a security parameter $k$ and generates the public parameter $Para = (\mathbb{G}_1, \mathbb{G}_T, p, P, e, H, H_1)$ by $Setup$ algorithm. Then $\mathcal{B}$ performs $KeyGen$ algorithm to generate the public/secret key pairs $(pk_s, sk_s) = (aP, a)$ and $(pk_u, sk_u) = (bP, b)$ respectively for data sender(DS) and data user(DU). Finally the challenger $\mathcal{B}$ makes $Para$ and public keys which is $pk_s$ and $pk_u$ known to the public.
- $Phase\ I$. The challenger $\mathcal{B}$ answers $\mathcal{A}$'s adaptive queries below in polynomial time:
  - Ciphertext query: The adversary $\mathcal{A}$ gives the challenger $\mathcal{B}$ a keyword $w$. Then $\mathcal{B}$ executes the encryption algorithm and returns the output $C_w$ which is the ciphertext of $w$ back to the adversary $\mathcal{A}$.
  - Trapdoor query: The adversary $\mathcal{A}$ gives the challenger $\mathcal{B}$ a keyword $w$. Then $\mathcal{B}$ executes the algorithm $Trapdoor$ and returns the output $T_w$ which is the trapdoor of $w$ back to the adversary $\mathcal{A}$.
  - Hash query: $\mathcal{B}$ first initializes an empty hash list $L_H$ for storing simulating hash value of hash function $H$. $\mathcal{A}$ gives $\mathcal{B}$ some element $P' \in \mathbb{G}_1$. $\mathcal{B}$ then checks if $P'$ is in $L_H$. If exists, return the hash value to $\mathcal{A}$. Otherwise, $\mathcal{B}$ picks a random number $v \in \mathbb{Z}_p^*$ as the hash value of $P'$ where $v$ is not duplicated in the hash values of $L_H$ then sets an entry $(P', v)$ and pushes the entry into $L_H$. Finally, $\mathcal{B}$ returns $v$ back to $\mathcal{A}$.
- $Challenge$. $\mathcal{A}$ chooses two keywords $w_0$, $w_1$ that have not been queried for trapdoors and sends them to the challenger $\mathcal{B}$. Then $\mathcal{B}$ randomly chooses $b \in \{0, 1\}$ and $r \in \mathbb{Z}_p^*$, to generate the challenge ciphertext $C_{w_b} = DA\text{-}SPKE(Para, pk_u, sk_s, w_b) = (C_{1w_b}, C_{2w_b}, C_{3w_b})$:

$$\begin{aligned}
C_{1w_b} &= rP \\
C_{2w_b} &= rabP \\
C_{3w_b} &= e(aP, bP)^{r H_1(w_b)^{H(abP)}}
\end{aligned}$$

- $Phase\ II$. This stage is the same as $Phase\ I$. While the only restriction is that neither $w_0$ nor $w_1$ should be queried to the trapdoor query. Also, the hash list used in the hash query should continue to be $L_H$ used in $Phase\ I$ rather than initialize a new one.
- $Guess$: $\mathcal{A}$ outputs her guess $b' \in \{0, 1\}$ for $b$ and wins the game if $b' = b$ with advantage $Adv_{\mathcal{A}}^{Game0}(k)$.

The advantage $\mathcal{A}$ gets in Game 0 is:
$Adv_{\mathcal{A}}^{Game0}(k) = Adv_{\mathcal{A}}^{IND\text{-}CKA}(k) = |Pr[b = b'] - \frac{1}{2}|$.

**Game 1**. Game 1 is almost the same as Game 0 except that when $\mathcal{B}$ generates the challenge ciphertext $C_{w_b}$ randomly chooses $T \in \mathbb{G}_1$ instead of computing $abP$ and uses $T$ instead of $abP$ to answer queries in both $Phase\ I$ and $Phase\ II$. The challenge ciphertext $C^*_{w_b} = (C^*_{1w_b}, C^*_{2w_b}, C^*_{3w_b})$ now becomes as below:

$$\begin{aligned}
C^*_{1w_b} &= rP \\
C^*_{2w_b} &= rT \\
C^*_{3w_b} &= e(T, P)^{r H_1(w_b)^{H(T)}}
\end{aligned}$$

In the case where $\mathcal{A}$ owns $(P, aP, bP)$, $\mathcal{A}$ is not able to compute $abP$ under the circumstances that the CDH assumption holds. Therefore from the adversary's point of view, she can't distinguish $C_{3w_b}$ and $C^*_{3w_b}$ as well as $C_{2w_b}$ and $C^*_{2w_b}$ in the random oracle model. Therefore for any PPT adversary $\mathcal{A}$, $C_{3w_b}$ and $C^*_{3w_b}$ as well as $C_{2w_b}$ and $C^*_{2w_b}$ are indistinguishable, then we have:

$$|Adv_{\mathcal{A}}^{Game1}(k) - Adv_{\mathcal{A}}^{Game0}(k)| \leq Adv_{\mathcal{A}}^{CDH}(k)$$

**Game 2**. Game 2 is almost the same as Game 1 except that when $\mathcal{B}$ generates the challenge ciphertext $C_{w_b}$ she randomly chooses $u \in \mathbb{Z}_p^*$ instead of computing $H(T)$ and uses $u$ instead of $H(T)$ to answer queries in both $Phase\ I$ and $Phase\ II$. The challenge ciphertext $C^{**}_{w_b} = (C^*_{1w_b}, C^*_{2w_b}, C^{**}_{3w_b})$ now becomes as below:

$$\begin{aligned}
C^*_{1w_b} &= rP \\
C^*_{2w_b} &= rT \\
C^{**}_{3w_b} &= e(T, P)^{r H_1(w_b)^u}
\end{aligned}$$

In the occasion where random oracle is used $u$ and $H(T)$ have the same distribution. Therefore from the adversary's point she is incapable of distinguishing $C^*_{3w_b}$ and $C^{**}_{3w_b}$ in the random oracle model. Therefore $\mathcal{A}$ can't distinguish $C^*_{w_b}$ and $C^{**}_{w_b}$, then we have:

$$Adv^{Game2}_{\mathcal{A}}(k) = Adv^{Game1}_{\mathcal{A}}(k)$$

**Game 3**. Game 3 is almost the same as Game 2 except that when $\mathcal{B}$ generates the challenge ciphertext $C^{***}_{w_b}$ randomly chooses $C^{***}_{1w_b}, C^{***}_{2w_b} \in \mathbb{G}$ and $C^{***}_{3w_b} \in \mathbb{G}_T$ instead of computing ciphertext. The challenge ciphertext then becomes $C^{***}_{w_b} = (C^{***}_{1w_b}, C^{***}_{2w_b}, C^{***}_{3w_b})$ .

Based on the randomness of $r$, $C^*_{1w_b}$ and $C^{***}_{1w_b}$ have the same distribution, $C^*_{2w_b}$ and $C^{***}_{2w_b}$ as well, for any PPT adversary $\mathcal{A}$. Due to the randomness of $u$, $\mathcal{A}$ is not able to distinguish $C^{**}_{3w_b}$ and $C^{***}_{3w_b}$ in the random oracle model. Then we have

$$Adv^{Game3}_{\mathcal{A}} = Adv^{Game2}_{\mathcal{A}}.$$

Because $C^{***}_{w_b}$ is independent of $b$, we have

$$Pr[b = b'] = \frac{1}{2}.$$

At this point, with the combination of **Game 0**, **Game 1**, **Game 2** and **Game 3**, we have

$$|Adv^{Game3}_{\mathcal{A}}(k) - Adv^{IND\text{-}CKA}_{\mathcal{A}}(k)| \leq Adv^{CDH}_{\mathcal{A}}(k)$$

where $Adv^{Game3}_{\mathcal{A}}(k) = 0$ and $Adv^{CDH}_{\mathcal{A}}(k)$ is negligible. So $Adv^{IND\text{-}CKA}_{\mathcal{A}}(k)$ is negligible.

**Remark**. In Game 1, we use $T$ instead of $abP$ for public key pair $aP$, $bP$. However, the adversary cannot tell them when $T$ is hiding. This is because the adversary cannot compute $abP$ from $(aP, bP)$ then obviously cannot compute $rabP$. It's impossible to check the equality $C^*_{2w_b} = rabP$ for the adversary. In Game 2, we use random value $u$ instead of $H(T)$, which leads to the distribution of the challenge ciphertext in Game 1 and Game 2 being identical under the CDH assumption in the random oracle model. Because $u$ is randomly chosen, which causes the distribution of $C^{**}_{3w_b}$ uniform distribution. Therefore, In Game 3, $C^{***}_{3w_b}$ is also randomly chosen instead of randomly choosing $u$ and then computing $e(T, P)^{rH(w_b)^u}$.

*Theorem 2:* Our improved scheme is IND-KGA security if the CDH assumption holds. In other words, for any PPT adversary $\mathcal{A}$, her advantage $Adv^{IND\text{-}KGA}_{\mathcal{A}}$ in IND-KGA security game is negligible.

**Proof**:We define four security games – **Game 0**, **Game 1**, **Game 2** and **Game 3** below.

**Game 0**.

- $Setup$. The challenger $\mathcal{B}$ selects a security parameter $k$ and generates the public parameter $Para = (\mathbb{G}_1, \mathbb{G}_T, p, P, e, H, H_1)$ by $Setup$ algorithm. Then $\mathcal{B}$ performs $KeyGen$ algorithm to generate the public/secret key pairs $(pk_s, sk_s) = (aP, a)$ and $(pk_u, sk_u) = (= bP, b)$ respectively for DS and DU. Finally the challenger $\mathcal{B}$ makes $Para$ and public keys which is $pk_s$ and $pk_u$ known to the public.
- $Phase\ I$. The challenger $\mathcal{B}$ answers $\mathcal{A}$'s adaptive queries below in polynomial time:
  - Ciphertext query: The adversary $\mathcal{A}$ gives the challenger $\mathcal{B}$ a keyword $w$. Then $\mathcal{B}$ executes the encryption algorithm and returns the output $C_w$ which is the ciphertext of $w$ back to the adversary $\mathcal{A}$.
  - Trapdoor query: The adversary $\mathcal{A}$ gives the challenger $\mathcal{B}$ a keyword $w$. Then $\mathcal{B}$ executes the algorithm $Trapdoor$ and returns the output $T_w$ which is the trapdoor of $w$ back to the adversary $\mathcal{A}$.
  - Hash query: $\mathcal{B}$ first initializes an empty hash list $L_H$ for storing simulating hash value of hash function $H$. $\mathcal{A}$ gives $\mathcal{B}$ some element $P' \in \mathbb{G}_1$. $\mathcal{B}$ then checks if $P'$ is in $L_H$. If exists, return the hash value to $\mathcal{A}$. Otherwise, $\mathcal{B}$ picks a random number $v \in \mathbb{Z}^*_p$ as the hash value of $P'$ where $v$ is not duplicated in the hash values of $L_H$ then sets an entry $(P', v)$ and pushes the entry into $L_H$. Finally, $\mathcal{B}$ returns $v$ back to $\mathcal{A}$.
- $Challenge$. $\mathcal{A}$ chooses two keywords $w_0, w_1$ that have not been queried for both ciphertext and trapdoor, and then sends them to the challenger $\mathcal{B}$. $\mathcal{B}$ randomly chooses $b \in \{0, 1\}$ and $r \in \mathbb{Z}^*_p$, to the challenge trapdoor $T_{w_b} = (T_{1w_b}, T_{2w_b})$:

$$\begin{aligned} T_{1w_b} &= rP \\ T_{2w_b} &= (r + H_1(w_b)^{H(sk_u pk_s)}) sk_u pk_s \\ &= (r + H_1(w_b)^{H(abP)}) abP. \end{aligned}$$

- $Phase\ II$. This stage is the same as $Phase\ I$. While the only restriction is that neither $w_0$ nor $w_1$ should be queried to ciphertext and trapdoor query. Also, the hash list used in the hash query should continue to be $L_H$ used in $Phase\ I$ rather than initialize a new one.
- $Guess$: $\mathcal{A}$ outputs her guess $b' \in \{0, 1\}$ and wins the game if $b' = b$ with advantage $Adv^{Game0}_{\mathcal{A}}$.

The advantage of $\mathcal{A}$ in Game 0 is:
$Adv^{Game0}_{\mathcal{A}}(k) = Adv^{IND\text{-}KGA}_{\mathcal{A}}(k) = |Pr[b = b'] - \frac{1}{2}|$.

**Game 1**. Game 1 is almost the same as Game 0 except that when $\mathcal{B}$ generates the challenge trapdoor $T_{w_b}$ randomly chooses $Q \in \mathbb{G}_1$ instead of computing $abP$ and uses $Q$ instead of $abP$ to answer queries in both $Phase\ I$ and $Phase\ II$. The challenge trapdoor $T^*_{w_b} = (T^*_{1w_b}, T^*_{2w_b})$ now becomes as below:

$$\begin{aligned} T^*_{1w_b} &= rP \\ T^*_{2w_b} &= rQ + H_1(w_b)^{H(Q)} Q \end{aligned}$$

In the case where $\mathcal{A}$ owns $(P, aP, bP)$, $\mathcal{A}$ is not able to compute $abP$ under the circumstances that the CDH assumption holds. Therefore for any PPT adversary $\mathcal{A}$, $T_{w_b}$ and $T^*_{w_b}$ are indistinguishable in the random oracle model, then we have:

$$|Adv^{Game1}_{\mathcal{A}}(k) - Adv^{Game0}_{\mathcal{A}}(k)| \leq Adv^{CDH}_{\mathcal{A}}(k)$$

**Game 2**. Game 2 is almost the same as Game 1 except that when $\mathcal{B}$ generates the challenge trapdoor $T_{w_b}$ randomly chooses $u \in \mathbb{Z}^*_p$ instead of computing $H(Q)$ and uses $u$ instead of $H(Q)$ to answer queries in both $Phase\ I$ and $Phase\ II$.

The challenge trapdoor $T_{w_b}^{**} = (T_{1w_b}^{*}, T_{2w_b}^{**})$ now becomes as below:

$$T_{1w_b}^{*} = rP$$
$$T_{2w_b}^{**} = rQ + H_1(w_b)^u Q$$

In the occasion where random oracle is used $u$ and $H(Q)$ have the same distribution. Therefore from the adversary's point she is incapable of distinguishing $T_{2w_b}^{*}$ and $T_{2w_b}^{**}$ in the random oracle model. Therefore $\mathcal{A}$ can't distinguish $T_{w_b}^{*}$ and $T_{w_b}^{**}$, then we have:

$$Adv_{\mathcal{A}}^{Game2}(k) = Adv_{\mathcal{A}}^{Game1}(k)$$

**Game 3**. Game 3 is almost the same as Game 2 except that when $\mathcal{B}$ generates the challenge trapdoor $T_{w_b}^{***}$ randomly chooses $T_{1w_b}^{***}, T_{2w_b}^{***} \in \mathbb{G}_1$ instead of computing trapdoor. The challenge trapdoor then becomes $T_{w_b}^{***} = (T_{1w_b}^{***}, T_{2w_b}^{***})$ .

Based on the randomness of $r$, for any PPT adversary $\mathcal{A}$, $T_{1w_b}^{*}$ and $T_{1w_b}^{***}$ are indistinguishable. Due to the randomness of $u$, $\mathcal{A}$ is not able to distinguish $T_{2w_b}^{**}$ and $T_{2w_b}^{***}$ in the random oracle model. Then we have: $Adv_{\mathcal{A}}^{Game3}(k) = Adv_{\mathcal{A}}^{Game2}(k)$.

Because $T_{w_b}^{***}$ is independent of $b$, we have:

$$Pr[b = b'] = \frac{1}{2}$$

At this point, with the combination of **Game 0**, **Game 1**, **Game 2** and **Game 3**, we have:

$$|Adv_{\mathcal{A}}^{Game3}(k) - Adv_{\mathcal{A}}^{IND\text{-}KGA}(k)| \leq Adv_{\mathcal{A}}^{CDH}(k)$$

where $Adv_{\mathcal{A}}^{Game3}(k) = 0$ and $Adv_{\mathcal{A}}^{CDH}(k)$ is negligible. So $Adv_{\mathcal{A}}^{IND\text{-}KGA}(k)$ is negligible.

**Remark**. Our IND-KGA secure model is derived from Wu et al's IND-KGA definition [27] but with a slight difference, where challenge keywords are forbidden to query in both ciphertext query and trapdoor query in the query phase rather than only ciphertext query. In Game 1, we use $Q$ instead of $abP$ for public key pair $aP$, $bP$. However, the adversary cannot tell them when $Q$ is behind the curtain. This is because the adversary cannot compute $abP$ from $(P, aP, bP)$ then check the equality $T_{2w_b}^{*} = (r + H_1(w_b)^{H(Q)})abP$. In Game 2, we use random value $u$ instead of $H(T)$, which leads the distribution of the challenge ciphertext in Game 1 and Game 2 identical under the CDH assumption in the random oracle model. Because $u$ is randomly chosen, which causes the distribution of $T_{2w_b}^{**}$ uniform. Therefore, In Game 3, $T_{2w_b}^{***}$ is also randomly chosen instead of randomly choosing $u$ and then computing $(r + H_1(w_b)^u)Q$.

### *E. Performance*

Analysis of the performance is in this section. To evaluate the efficiency of the proposed DA-SPKE scheme, We will firstly compare the computation cost and the communication cost in theory to prior similar schemes [25]–[31], then we will display the data in real experiments with selected schemes [25], [27], [31] for computation cost. For the convenience of description, we denote every scheme by the last name of the first author, that is we denote [25] by Boneh, [26] by Rhee, [27] by Wu, [28] by Xu, [29] by Huang, [30] by Zeng and [31] by Qin respectively.

The running environment is a Macbook Pro with MacOS Ventura operating system, where the processor is Apple M1 silicon using 8GB RAM, we finish the code by Java of jdk17 with the use of bilinear pairing library JPBC library with the Type A pairing over elliptic curve $y^2 = x^3 + x$. In the experiment, we ran every scheme 10,000 times and got the average time for every operation as shown in Table II, where notations for every basic operation and their descriptions are defined.

We firstly mention the functionality of our proposed scheme as shown in Table III. The security of schemes is measured in terms of IND-CKA, the IND-KGA, authentication, deniability, and secure channel, where IND-CKA security denotes that the ciphertext of the keyword does not leak any information about the keyword, IND-KGA shows that the trapdoor of keyword does not reveal any information about the keyword, authentication denotes that only the authenticated user can generate a legal trapdoor to search the ciphertext of keywords and secure channel means that the scheme requires a secure channel to transmit the trapdoor. Our scheme has the most secure features among all the similar prior works.

We then compare the computation cost and communication cost separately in theory with prior similar schemes. The details are shown in Table IV and Table V respectively, where $E, P, H$ denotes the multiplication over the addictive group, bilinear pairing and the hash function mapping from $\mathbb{Z}_p^*$ to group $\mathbb{G}_1$, and $|p|, |\mathbb{G}_1|, |\mathbb{G}_T|$ denotes the length of elements from $\mathbb{Z}_p^*, \mathbb{G}_1$ and $\mathbb{G}_T$ respectively.

In the computation cost comparison, we mainly consider the costs of the encryption algorithm $encryption$, the trapdoor generation algorithm $trapdoor$, and the match testing algorithm $test$. As for other algorithms such as set up algorithm $Setup$, and key generation algorithm $KeyGen$, they are usually executed once and beforehand by an entity. Two deniable algorithms perform the same as the normal two, the reason why is explained in Section III-B. Therefore, these algorithms are not involved in the comparison.

The theoretical computation cost of an algorithm is evaluated by the sum of the number of basic operations involved in the algorithm. For example, to produce a ciphertext, the algorithm $encryption$ in our scheme requires calculating four scalar multiplications in the elliptic curve group, one bilinear pairing over the elliptic curve group. Thus, its computation cost is $4E + P \approx 28.225$ ms. In addition, the size of a public/secret key, ciphertext, or trapdoor is measured by the total number of the involved elements. For example, a ciphertext in our scheme contains two elements in $\mathbb{G}_1$ and one element in $\mathbb{G}_T$. Thus, its size is $2|\mathbb{G}_1| + |\mathbb{G}_T| = 1152$ bits.

The experimental results of time costs of four schemes are shown in Fig. 2, where the horizontal axis represents the algorithm execution times while the vertical axis represents the corresponding time cost. We respectively execute the algorithms in each compared scheme 1 to 10 times to show their time costs. Our scheme gains great performance computationally compared to other similar schemes, which

TABLE II
NOTATIONS IN THE PERFORMANCE EVALUATION

| | Notations | Performance | Descriptions |
|---|---|---|---|
| Execution Time (ms) | $E$ | 5.914 | execution time of multiplication over addictive group |
| | $P$ | 4.569 | execution time of bilinear pairing |
| | $H$ | 6.586 | execution time of hash function mapping from $\mathbb{Z}_p^*$ to group $\mathbb{G}_1$ |
| Bit Length (bit) | $\|p\|$ | 160 | the length of elements from $\mathbb{Z}_p^*$ |
| | $\|\mathbb{G}_1\|$ | 320 | the length of elements from $\mathbb{G}_1$ |
| | $\|\mathbb{G}_T\|$ | 512 | the length of elements from $\mathbb{G}_T$ |

TABLE III
SECURITY COMPARISON

| **Scheme** | **CKA** | **KGA** | **Authentication** | **Deniability** | **Secure Chanel** |
|---|---|---|---|---|---|
| [25] | yes | no | no | no | yes |
| [26] | yes | yes | no | no | no |
| [27] | yes | no | yes | yes | yes |
| [28] | yes | yes | no | no | no |
| [29] | yes | yes | no | no | no |
| [31] | yes | yes | yes | no | no |
| our scheme | yes | yes | yes | yes | no |

TABLE IV
COMPUTATION COST EVALUATION (MS) WITH PRIOR WORKS

| Scheme | encryption | trapdoor | test |
|---|---|---|---|
| [25] | $2E + P + H \approx 22.983$ | $E + H \approx 12.450$ | $P \approx 4.569$ |
| [26] | $2E + P + H \approx 22.983$ | $3E + 2H \approx 30.913$ | $2E + P + H \approx 22.983$ |
| [27] | $4E + P \approx 28.225$ | $4E \approx 23.656$ | $2P \approx 9.139$ |
| [28] | $5E + 2P + 2H \approx 51.880$ | $3E + 2H \approx 30.913$ | $2P \approx 9.139$ |
| [29] | $3E + H \approx 24.328$ | $E + P + H \approx 17.069$ | $2P \approx 9.139$ |
| [30] | $6E + H \approx 42.069$ | $8E + H \approx 53.897$ | $5P \approx 22.847$ |
| [31] | $3E + P + H \approx 28.897$ | $2E + H \approx 18.414$ | $P \approx 4.569$ |
| our scheme | $4E + P \approx 28.225$ | $3E \approx 17.742$ | $2P \approx 9.139$ |

TABLE V
COMMUNICATION COST EVALUATION (BIT) WITH PRIOR WORKS

| Scheme | pk | sk | ciphertext | trapdoor |
|---|---|---|---|---|
| [25] | $2\|\mathbb{G}_1\| = 640$ | $\|p\| = 160$ | $\|\mathbb{G}_1\| + \|p\| = 480$ | $\|\mathbb{G}_1\| = 320$ |
| [26] | $2\|\mathbb{G}_1\| = 640$ | $\|p\| = 160$ | $\|\mathbb{G}_1\| + \|p\| = 480$ | $2\|\mathbb{G}_1\| = 640$ |
| [27] | $\|\mathbb{G}_1\| = 320$ | $\|p\| = 160$ | $2\|\mathbb{G}_1\| + \|\mathbb{G}_T\| = 832$ | $2\|\mathbb{G}_1\| = 640$ |
| [28] | $\|\mathbb{G}_1\| = 320$ | $\|p\| = 160$ | $2\|\mathbb{G}_1\| = 640$ | $2\|\mathbb{G}_1\| + 2\|p\| = 960$ |
| [29] | $\|\mathbb{G}_1\| = 320$ | $\|p\| = 160$ | $2\|\mathbb{G}_1\| = 640$ | $\|\mathbb{G}_T\| = 512$ |
| [30] | $4\|\mathbb{G}_1\| = 1280$ | $3\|p\| = 480$ | $4\|\mathbb{G}_1\| = 1280$ | $5\|\mathbb{G}_1\| = 1600$ |
| [31] | $\|\mathbb{G}_1\| = 320$ | $\|p\| = 160$ | $\|\mathbb{G}_1\| + \|p\| = 480$ | $\|\mathbb{G}_1\| = 320$ |
| our scheme | $\|\mathbb{G}_1\| = 320$ | $\|p\| = 160$ | $2\|\mathbb{G}_1\| + \|\mathbb{G}_T\| = 1152$ | $2\|\mathbb{G}_1\| = 640$ |

lower than average computation costs. More specifically, the time cost for $encryption$ in our scheme is about 28.225 ms, which is about 54.4 percent of 51.880 ms in Xu [28] and 67.1 percent of 42.069 ms in Zeng [30]. The time cost for $trapdoor$ in our scheme is about 17.742 ms, which is about 57.4 percent of 30.913 ms in Rhee [26] and Xu [28], 32.9 percent of 53.897 ms in Zeng [30] and 75 percent of 23.656 ms in Wu [27]. The time cost for $test$ in our scheme is about 9.139 ms, which is about 39.8 percent of 22.983 ms in Rhee [26], and 40 percent of 22.847 ms in Zeng [30].

Regarding the communication overhead as illustrated in Fig. 3, the size of a public key $pk$ in our scheme is 320 bits, while that in Boneh [25] and Rhee [26] is 640 bits, Wu [27], Xu [28], Huang [29] and Qin [31] is 320 bits, and Zeng [30] is 1280 bits. The size of a secret key $sk$ in our scheme is 160 bits the same as most of the schemes except that Zeng [30] has a length of 480 bits. The size of a ciphertext $ciphertext$ in our scheme is 1152 bits, which is 90 percent of 1280 bits in Zeng [30]. The size of a trapdoor $trapdoor$ in our scheme is 640 bits, while that in Boneh, Rhee, Wu, Xu, Huang, Zeng, and Qin is 320 bits, 640 bits, 640 bits, 960 bits, 512 bits, 1600 bits, and 320 bits, respectively. Therefore our scheme gains great communication performance among the compared schemes.

## IV. SECURE EMAIL SEARCHABLE KEYWORD PROTOCOL

In this section, we design a secure email searchable keyword protocol using the proposed deniable authenticated searchable

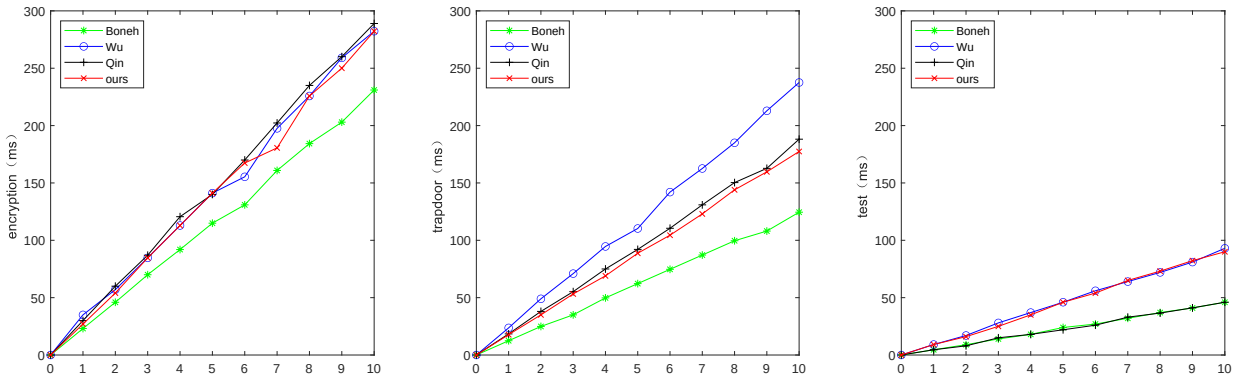



Fig. 2. Selected Computation Cost Comparison

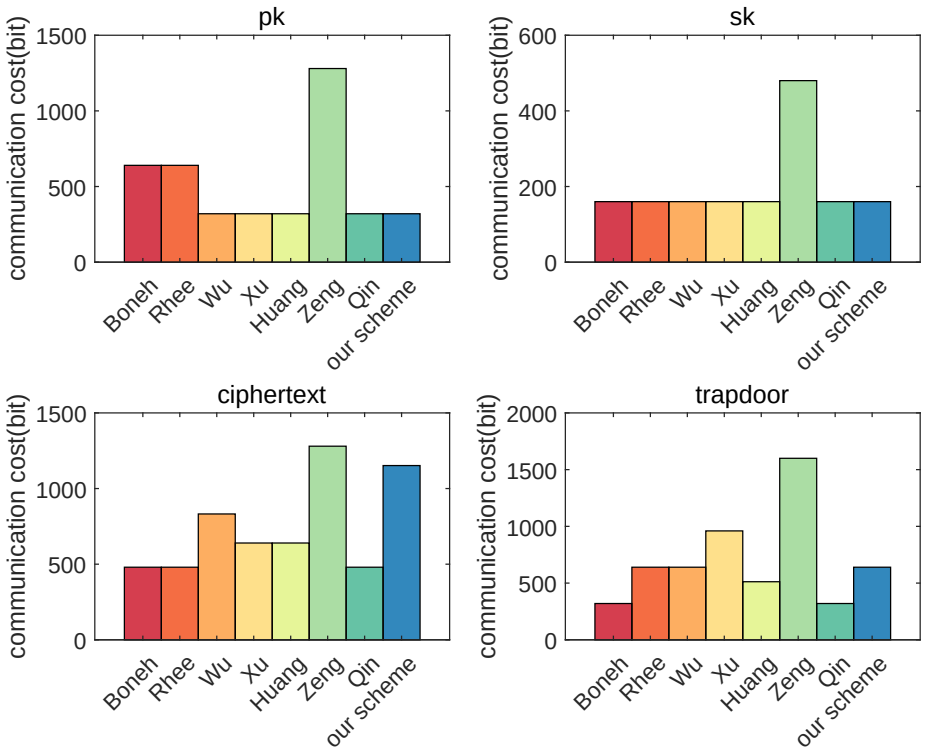



Fig. 3. Communication Cost Comparison

public key encryption scheme. The protocol is shown in Fig. 4. It is worth mentioning that the content of the email is encrypted by some symmetric encryption scheme which is denoted by $sym\text{-}enc$ such as AES and 3DES for the reason of high speed. The secret keys to encrypt and decrypt the content are the same. So the secret key should be kept secret by both the authorized data sender and data user.

In this secure protocol, the data sender firstly chooses some keywords $\{w_i\}_{i\in[m]}$ for his email, then he runs $c_i = DA\text{-}SPKE(Para, pk_u, sk_s, w_i)$ to obtain the ciphertext set $\{c_i\}_{i\in[m]}$, where $[m]$ denotes $\{1, 2, ..., m\}$. After that, the data sender encrypts the content of the email using the secret key of the symmetric encryption scheme. Then the data sender sends the encrypted email along with $\{c_i\}_{i\in[m]}$ to the email server by Simple Mail Transfer Protocol (SMTP), which is a common email transfer protocol. The email server sends the encrypted data to the cloud by SMTP. When the authorized data user wants to search for a keyword $w$ which is encrypted in the email system, he runs $td = Trapdoor(Para, pk_s, sk_u, w)$ to obtain trapdoor $td$ of $w$. Then the data user sends $td$ to the email server by SMTP. The email server sends $td$ to the cloud. The cloud runs $Test(Para, td, c_i)$ to traverse retrieval of every encrypted keyword to get the matched email in the cloud database (as shown in Fig. 5). Then the cloud sends the matched email along with the encrypted keywords to the email server. Finally, the data user receives the email by Post Office Protocol-Version 3 (POP3), where POP3 is a common email transfer protocol for the email receiver.

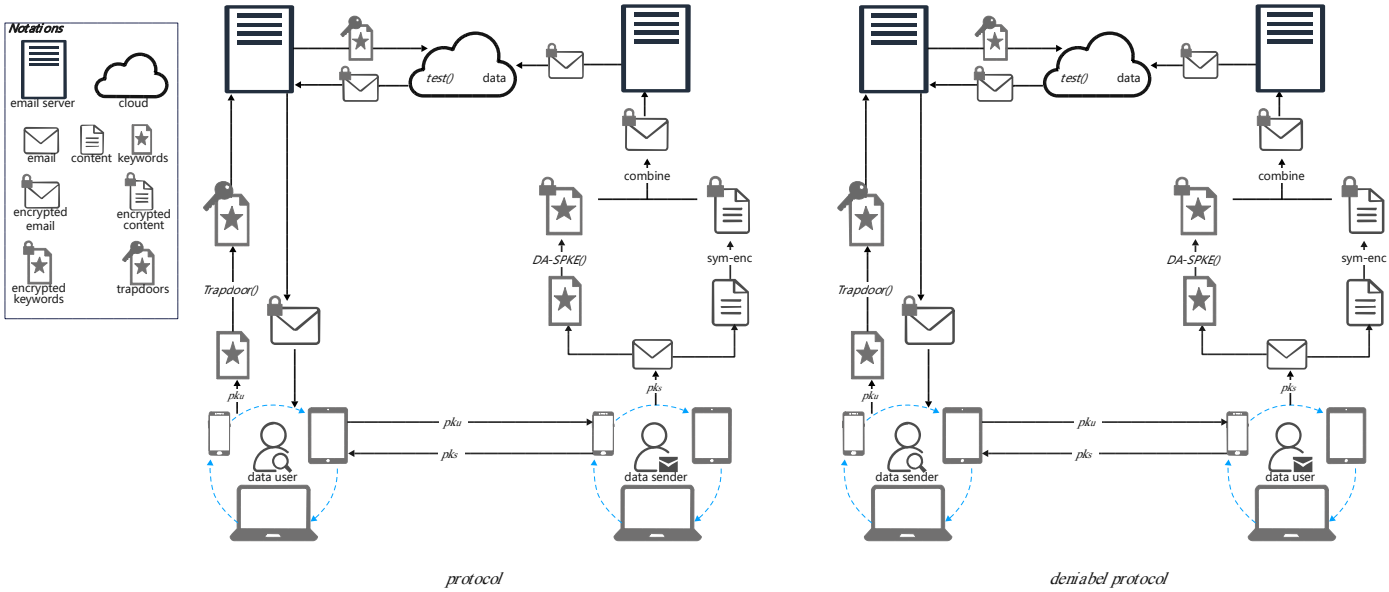



Fig. 4. protocol.

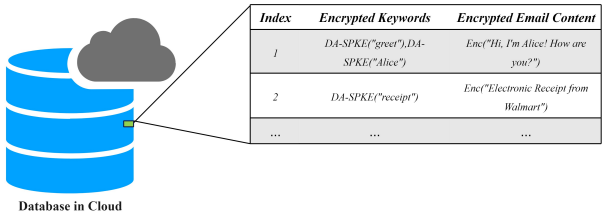



Fig. 5. Database in Cloud

In this protocol, we cut down the computation and storage expenses in the transfer protocol, which means there are no extra encryption expenses while transferring email data. We propose a new approach to search encrypted keywords in a secure email system, where the proposed scheme firstly offers deniability for the users of the system shown in Fig. 4. In the deniable scene, the only difference is the exchange of roles, the data sender can produce a valid trapdoor and the data user can produce a valid ciphertext by deniable algorithms. In this case, the users in the email system can better protect their privacy as they are anonymous in a sense.

## V. Conclusion

In our work, we initially proposed a deniably authenticated searchable public key encryption that is deniable both in keyword encryption and trapdoor generation in the email system scenario. After that, we prove our scheme to be both IND-KGA and IND-CKA secure. Then, Compared with other solutions, our performance has a performance advantage. Finally, we proposed a secure email searchable keyword protocol using the proposed deniable scheme.

## Acknowledgments

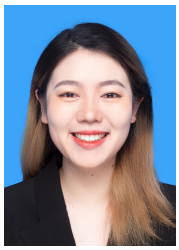
This study was funded by the Sichuan Science and Technology Program under Grant 2023YFG0031.

## References

[1] S. Garfinkel, *PGP: pretty good privacy*. " O'Reilly Media, Inc.", 1995.
[2] B. Ramsdell and S. Turner, "Secure/multipurpose internet mail extensions (s/mime) version 3.2 message specification," Tech. Rep., 2010.
[3] Z. Liu and Y. Liu, "Verifiable and authenticated searchable encryption scheme with aggregate key in cloud storage," in *2018 14th International Conference on Computational Intelligence and Security (CIS)*. IEEE, 2018, pp. 421–425.
[4] M. I. Mihailescu and S. L. Nita, "A searchable encryption scheme with biometric authentication and authorization for cloud environments," *Cryptography*, vol. 6, no. 1, p. 8, 2022.
[5] L. Chen, K. Huang, M. Manulis, and V. Sekar, "Password-authenticated searchable encryption," *International Journal of Information Security*, vol. 20, pp. 675–693, 2021.
[6] L. Cheng and F. Meng, "Certificateless public key authenticated searchable encryption with enhanced security model in iiot applications," *IEEE Internet of Things Journal*, vol. 10, no. 2, pp. 1391–1400, 2022.
[7] C. Xu, M. Lin, J. Cheng, Y. Zhao, and C. Zuo, "Iot services: realizing private real-time detection via authenticated conjunctive searchable encryption," *Journal of Cybersecurity*, vol. 3, no. 1, p. 55, 2021.
[8] J. Kar, K. Naik, and T. Abdelkader, "An efficient and lightweight deniably authenticated encryption scheme for e-mail security," *IEEE Access*, vol. 7, pp. 184 207–184 220, 2019.
[9] M. AlSabah, A. Tomescu, I. Lebedev, D. Serpanos, and S. Devadas, "Privipk: Certificate-less and secure email communication," *Computers & Security*, vol. 70, pp. 1–15, 2017.
[10] D. X. Song, D. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," in *Proceeding 2000 IEEE symposium on security and privacy. S&P 2000*. IEEE, 2000, pp. 44–55.
[11] A. J. Aviv, M. E. Locasto, S. Potter, and A. D. Keromytis, "Ssares: Secure searchable automated remote email storage," in *Twenty-Third Annual Computer Security Applications Conference (ACSAC 2007)*. IEEE, 2007, pp. 129–139.
[12] W. Zhang, B. Qin, X. Dong, and A. Tian, "Public-key encryption with bidirectional keyword search and its application to encrypted emails," *Computer Standards & Interfaces*, vol. 78, p. 103542, 2021.

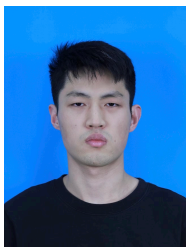
[13] P. Xu, S. Tang, P. Xu, Q. Wu, H. Hu, and W. Susilo, "Practical multi-keyword and boolean search over encrypted e-mail in cloud server," *IEEE Transactions on Services Computing*, vol. 14, no. 6, pp. 1877–1889, 2019.
[14] Q. Huang and H. Li, "An efficient public-key searchable encryption scheme secure against inside keyword guessing attacks," *Information Sciences*, vol. 403, pp. 1–14, 2017.
[15] W.-C. Yau, R. C.-W. Phan, S.-H. Heng, and B.-M. Goi, "Keyword guessing attacks on secure searchable public key encryption schemes with a designated tester," *International Journal of Computer Mathematics*, vol. 90, no. 12, pp. 2581–2587, 2013.
[16] Y. Lu and J. Li, "Efficient searchable public key encryption against keyword guessing attacks for cloud-based emr systems," *Cluster Computing*, vol. 22, pp. 285–299, 2019.
[17] M. Noroozi and Z. Eslami, "Public-key encryption with keyword search: a generic construction secure against online and offline keyword guessing attacks," *Journal of Ambient Intelligence and Humanized Computing*, vol. 11, pp. 879–890, 2020.
[18] B. Wu, C. Wang, and H. Yao, "Security analysis and secure channel-free certificateless searchable public key authenticated encryption for a cloud-based internet of things," *PloS one*, vol. 15, no. 4, p. e0230722, 2020.
[19] Z.-Y. Liu, Y.-F. Tseng, R. Tso, M. Mambo, and Y.-C. Chen, "Public-key authenticated encryption with keyword search: Cryptanalysis, enhanced security, and quantum-resistant instantiation," in *Proceedings of the 2022 ACM on Asia conference on computer and communications security*, 2022, pp. 423–436.
[20] Z. Jiang, K. Zhang, L. Wang, and J. Ning, "Forward secure public-key authenticated encryption with conjunctive keyword search," *The Computer Journal*, vol. 66, no. 9, pp. 2265–2278, 2023.
[21] K. Emura, "Generic construction of public-key authenticated encryption with keyword search revisited: stronger security and efficient construction," in *Proceedings of the 9th ACM on ASIA Public-Key Cryptography Workshop*, 2022, pp. 39–49.
[22] F. Li, Z. Zheng, and C. Jin, "Identity-based deniable authenticated encryption and its application to e-mail system," *Telecommunication Systems*, vol. 62, pp. 625–639, 2016.
[23] W. Huang, Y. Liao, S. Zhou, and H. Chen, "An efficient deniable authenticated encryption scheme for privacy protection," *IEEE Access*, vol. 7, pp. 43 453–43 461, 2019.
[24] Y. Cao, J. Wei, F. Zhang, Y. Xiang, and X. Chen, "Efficient public-key authenticated deniable encryption schemes," *Computer Standards & Interfaces*, vol. 82, p. 103620, 2022.
[25] D. Boneh, G. Di Crescenzo, R. Ostrovsky, and G. Persiano, "Public key encryption with keyword search," in *Advances in Cryptology-EUROCRYPT 2004: International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, May 2-6, 2004. Proceedings 23*. Springer, 2004, pp. 506–522.
[26] H. S. Rhee, J. H. Park, W. Susilo, and D. H. Lee, "Trapdoor security in a searchable public-key encryption scheme with a designated tester," *Journal of Systems and Software*, vol. 83, no. 5, pp. 763–771, 2010.
[27] L. Wu, B. Chen, S. Zeadally, and D. He, "An efficient and secure searchable public key encryption scheme with privacy protection for cloud storage," *Soft Computing*, vol. 22, pp. 7685–7696, 2018.
[28] P. Xu, H. Jin, Q. Wu, and W. Wang, "Public-key encryption with fuzzy keyword search: A provably secure scheme under keyword guessing attack," *IEEE Transactions on computers*, vol. 62, no. 11, pp. 2266–2277, 2012.
[29] Q. Huang and H. Li, "An efficient public-key searchable encryption scheme secure against inside keyword guessing attacks," *Information Sciences*, vol. 403, pp. 1–14, 2017.
[30] M. Zeng, H. Qian, J. Chen, and K. Zhang, "Forward secure public key encryption with keyword search for outsourced cloud storage," *IEEE transactions on cloud computing*, vol. 10, no. 1, pp. 426–438, 2019.
[31] B. Qin, Y. Chen, Q. Huang, X. Liu, and D. Zheng, "Public-key authenticated encryption with keyword search revisited: Security model and constructions," *Information Sciences*, vol. 516, pp. 515–528, 2020.
[32] W.-C. Yau, S.-H. Heng, and B.-M. Goi, "Off-line keyword guessing attacks on recent public key encryption with keyword search schemes," in *Autonomic and Trusted Computing: 5th International Conference, ATC 2008, Oslo, Norway, June 23-25, 2008 Proceedings 5*. Springer, 2008, pp. 100–105.
[33] P. Xu, S. He, W. Wang, W. Susilo, and H. Jin, "Lightweight searchable public-key encryption for cloud-assisted wireless sensor networks," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 8, pp. 3712–3723, 2017.
[34] B. Chen, L. Wu, H. Wang, L. Zhou, and D. He, "A blockchain-based searchable public-key encryption with forward and backward privacy for cloud-assisted vehicular social networks," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 6, pp. 5813–5825, 2019.
[35] M. Ma, D. He, N. Kumar, K.-K. R. Choo, and J. Chen, "Certificateless searchable public key encryption scheme for industrial internet of things," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 2, pp. 759–767, 2017.
[36] M. Ma, D. He, M. K. Khan, and J. Chen, "Certificateless searchable public key encryption scheme for mobile healthcare system," *Computers & Electrical Engineering*, vol. 65, pp. 413–424, 2018.
[37] Y. Lu and J. Li, "Lightweight public key authenticated encryption with keyword search against adaptively-chosen-targets adversaries for mobile devices," *IEEE Transactions on Mobile Computing*, vol. 21, no. 12, pp. 4397–4409, 2021.
[38] Y. Miao, F. Li, X. Li, Z. Liu, J. Ning, H. Li, K.-K. R. Choo, and R. H. Deng, "Time-controllable keyword search scheme with efficient revocation in mobile e-health cloud," *IEEE Transactions on Mobile Computing*, 2023.
[39] Q. Liu, Y. Peng, H. Jiang, J. Wu, T. Wang, T. Peng, and G. Wang, "Authorized keyword search on mobile devices in secure data outsourcing," *IEEE Transactions on Mobile Computing*, 2023.
[40] Y.-l. Zhang, L. Wen, Y.-j. Zhang, and C.-f. Wang, "Deniably authenticated searchable encryption scheme based on blockchain for medical image data sharing," *Multimedia Tools and Applications*, vol. 79, pp. 27 075–27 090, 2020.
[41] M. Hellman, "New directions in cryptography," *IEEE transactions on Information Theory*, vol. 22, no. 6, pp. 644–654, 1976.
[42] Q. Huang and H. Li, "An efficient public-key searchable encryption scheme secure against inside keyword guessing attacks," *Information Sciences*, vol. 403, pp. 1–14, 2017.

## VI. Biography Section

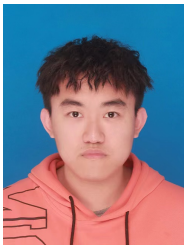


**Shuhan Zeng** received the B.E. degrees from the University of Electronic Science and Technology of China (UESTC), Chengdu, China, in 2023, respectively, where she is currently pursuing the M.S. degree with the School of Information and Software Engineering. Her research interests include searchable public key encryption and cybersecurity.

**Yongjian Liao** is currently an associate professor of School of Information and Software Engineering, University of Electronic Science and Technology of China. He received his Ph.D. degree in applied electronic science and technology from College of Information Science and Electronic Engineering, Zhejiang University in 2007. His main research interests include public key cryptography and information security, in particular, cryptographic protocols. He is a member of the IEEE.



**Chuanhao Zhou** received the B.E. degrees from the University of Electronic Science and Technology of China (UESTC), Chengdu, China, in 2020, respectively, where he received the M.S. degree with the School of Information and Software Engineering. His research interests include searchable public key encryption.



**Jinlin He** received the B.E. degrees from the University of Electronic Science and Technology of China (UESTC), Chengdu, China, in 2022, respectively, where he is currently pursuing the M.S. degree with the School of Information and Software Engineering. His research interests include blind signature (BS) and cybersecurity.

**Hongwei Wang** received the B.E. degrees from the University of Electronic Science and Technology of China (UESTC), Chengdu, China, in 2022, respectively, where he is currently pursuing the M.S. degree with the School of Information and Software Engineering. His research interests include identity-based encryption (IBE) and cybersecurity.