Encryption Switching Service: Securely Switch Your Encrypted Data to Another Format

Peng Jiang, Jianting Ning, Kaitai Liang, *Member, IEEE*, Changyu Dong, Jiageng Chen and Zhenfu Cao, *Senior Member, IEEE*

Abstract—Big data analytics has been regarded as a promising technology to yield better insights into future development by government and industry. Data collection and aggregation are necessary pre-steps to enable data analysis. However, data may be dispersed across multiple places and in different formats. Even worse, data can be encrypted under various encryption mechanisms when data owners try to secure the confidentiality of the data. This makes data aggregation extremely challenging, if not impossible, especially when the encryption keys cannot be shared for various reasons. In this paper, we take the first step in addressing this problem. More specifically, we propose a new notion of cross-domain encryption switching service that securely bridges two well-studied encryption mechanisms, namely traditional public key encryption and identity-based encryption. As of independent interest, our notion supports keyword search over encrypted data, i.e. after encryption switching one may search over the (outsourced) data without loss of data and query secrecy. We provide a provably-secure instantiation satisfying the notion, and further present the efficiency analysis to show the scalability. Our proposed scheme may be applicable in multi-domain cloud storage system.

Index Terms—Cross-domain encryption switching service, keyword search service, data secrecy, search privacy.

1 Introduction

End-to-end encryption has been seen as one of the secure solutions to protect the confidentiality of sensitive data while the data is outsourced and transferred in open network, like some of the real-world applications (e.g, cloud-based data storage and retrieval). The fast-paced growth of cloud-based applications urges the need of the flexibility of encryption, especially in the era of big data. Proxy re-encryption (PRE) [1], as a useful extension of public key encryption (PKE), formalizes an elegant approach to allow a semi-trusted proxy to transform ciphertexts among different data users. Given a re-encryption key, the proxy can convert a ciphertext intended for Alice to Bob, while maintaining the secrecy of the underlying plaintext and secret keys of both parties. Being seen as a ciphertext switching service, PRE has many practical applications, such as digital right management [2] and secure email forwarding [1]. For instance, manager Alice delegates her email decryption rights to secretary Helen by uploading a special key to email server before her annual leave, so that all Alice's encrypted email received during the leave can be gained access and read by Helen with Helen's decryption key. However, most of the existing PRE services deal with the conversion of ciphertexts under the same type of encryption, for example, an attribute-based encryption [3] can be only converted to the "same" encrypted format, which makes sense if this conversion service is only required within the same encryption system.

There are scenarios that cross-system ciphertext conversion service is desirable. Consider a health-care agent attempts to tailor a therapy plan for a patient. To do so, the agent needs to collect data of the patient (e.g., electronic health record, mental health report, employment data) from various data sources (belonging to different authorities), such as hospital, medical centre and even gym centre. But the data may be encrypted and stored in remote cloud and meanwhile, data is encrypted under different encryption systems. A naive way for the agent to "aggregate" the data (in which he is interested) is to request the corresponding decryption keys from the respective data handlers. This, however, yields concerns on data privacy. The data handlers may question: *"If I give you the key, how could you guarantee that the key will not be further leaked to others or be further used for other purposes?"*

This paper attempts to tackle the dilemma technically so that the agent can convert ciphertext from one encryption system to another without requesting secret keys from the data handlers. In particular, this paper considers the conversion service between two types of well-studied encryption systems, PKE and identity-based encryption (IBE). We here take medical record sharing service as a motivation of this work.

Assume *Hospital X* and *Hospital Y* use different policy rules (*Rule PKE* and *Rule IBE*) to handle their patients' medical data, and the data is stored with encrypted form in cloud. Since patient transfer is common case among hospitals. We suppose there is a patient that needs to be transferred from *Hospital X* to *Hospital Y*. Right after patient transfer, the electronic health record of the patient must be also delivered to *Hospital Y* immediately. If the medical data

P. Jiang is with School of Computer Science and Technology, Beijing Institute of Technology, Beijing, China. (e-mail: pennyjiang0301@gmail.com).
J. Ning is with Department of Computer Science, National University of Singapore, Singapore. (e-mail: jtning88@gmail.com. Corresponding author).
K. Liang is with Department of Computer Science, University of Surrey, U.K. (e-mail: k.liang@surrey.ac.uk).
C. Dong is with School of Computing Science, Newcastle University, U.K. (e-mail: changyu.dong@newcastle.ac.uk).
J. Chen is with Computer School, Central China Normal University, Wuhan 430079, China. (e-mail: jiageng.chen@mail.ccnu.edu.cn).
Z. Cao is with Shanghai Key Lab for Trustworthy Computing, East China Normal University, Shanghai, China (e-mail: zfcao@sei.ecnu.edu.cn).

is not encrypted, it can be shared between the hospitals easily (see Fig. 1). While the record is encrypted under *Rule PKE*, a direct access to the data is impossible for *Hospital Y*. Meanwhile, the encryption mechanism used by *Hospital X* and the corresponding decryption key may be unknown to *Hospital Y*. This leads *Hospital Y* to a question "how could it gain access to the patient's record?".

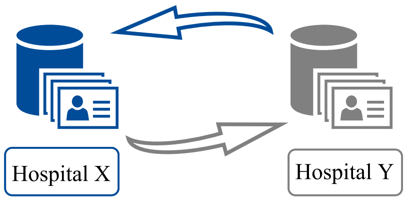


Fig. 1: Electronic Medical Record Sharing Among Hospitals.

Naive Solutions. One may leverage a trivial "decrypt-then-re-encrypt" solution in the sense that a trusted third party, holding *Hospital X* decryption key, may decrypt the record on behalf of *Hospital X* and then re-encrypt it under *Rule IBE* so that *Hospital Y* can read the record. However, this solution directly leaks the record to party, which is not a real privacy-preserving solution.

PRE would possibly come to help here. But the existing bidirectional PRE schemes (e.g., [4]) are defined with a restriction that the conversion can only be done with the input/output ciphertexts under the same encryption system. That is, *Hospital X* and *Hospital Y* must share the same policy rule. Accordingly, bidirectional PRE is infeasible solution since *Rule PKE* $\neq$ *Rule IBE* in our scenario.

As of independent interest, patient record archiving and retrieval are basic needs for hospital. For example, *Hospital Y* may first locate the patient's record before decrypting. It is necessary to maintain secure data search while supporting encryption switching service. One may think that trivially twisting PRE with public key encryption with keyword search (PEKS) [5] could become a handy solution. However, the straightforward combination makes two parts (the encrypted data and search part) loosely independent such that malicious cloud server can replace either of the parts with dummy component (which may lead to "always failure of search match"). Accordingly, *Hospital Y* may not retrieve any data with predefined keywords from cloud, which jeopardizes data usability.

This paper targets to tackle the following problem: *"How to provide cross-domain encryption switching service, between PKE and IBE, while supporting secure data search service."*

Contributions. In this work, we conduct the first study on a brand new cloud-based service - how to achieve cross-domain encryption switching with keyword search. Our study spans both bidirectional ciphertext transformation and searchable encryption perspectives for completeness. Inspired by the features of PRE service and public key encryption with keyword search (PEKS), we introduce a new notion, which is called *C̲ross-D̲omain Encryption S̲witching with Keyword S̲earch (CDSS)*. The notion is a bidirectional conversion bridge between PKE and IBE and meanwhile, the conversion maintains searchability. In particular, a PKE ciphertext can be transformed into the IBE one, and further be decrypted with the corresponding IBE decryption algorithm, and vice versa. Besides, all ciphertext formats (IBE, PKE and switching ciphertexts) support keyword search query. Our main contributions are summarized as follows.

- We, for the first time, define the notion (algorithm definition and security model) of cross-domain encryption switching with keyword search.
- We construct a concrete CDSS protocol between a PKE scheme and an IBE scheme, which satisfies the new notion.
- We define the chosen plaintext security and chosen keyword security models for the notion, and further prove the security of the presented protocol in random oracle model.
- We simulate our protocol on a file system. When the number of files is up to 6000, the time to generate the switching key and the ciphertext is about 70 seconds and 100 seconds, respectively. The efficiency analysis (see Section 8) shows both computational cost and communication cost of our CDSS protocol.

Applications. We state that our new design CDSS can be applied to many real-world cloud-based service applications. First of all, the example we use previously, the encrypted electronic health record sharing, is an appropriate area where the CDSS may be employed to. Encrypted email forwarding may be another practical application. Imagine that an encrypted email needs to go through two gateways for a successful delivery, in which the gateways are equipped with different encryption mechanisms. CDSS could be used as a compiler to smoothly turn an incoming encrypted message to the format matching the requirement of the entrance gateway. Digital contents transfer is also a potential application for CDSS. While transferring ownership (of digital property) from one party to another one, the legal representative of the original owner may make use of CDSS to fulfil secure content delivery without "read" the underlying contents. Last but not least, CDSS may be used as a plug-in for browser to display encrypted web content in the case where web content is encrypted under a standard encryption but web user customizes a different encryption mechanism for local browser.

2 Related Work

Switching Protocol. The primitive of encryption switching protocol (ESP) was introduced by Couteau et al. [6] to allow two players to interactively and obliviously convert an encryption to the other, for any polynomial number of switches, in any direction. Castagnos et al. [7] further improved and instantiated ESP based on Couteau et al.'s solid work. ESP is essentially a two-party computation protocol by leveraging cryptographic tools (e.g., homomorphic schemes). ESP makes use of shared decryption keys to obliviously decrypt and re-encrypt under the other encryption scheme, with a similar public key. ESP does not conform our scenario which requires a switch party to convert a ciphertext with a switch key. Also, ESP achieved an encryption switching without the property of search.

Inter/Cross-domain encryption. Blaze, Bleumer and Strauss [1] introduced the concept of PRE to tackle inter-domain ciphertext alteration, so that a new altered ciphertext may be decrypted by other users. PRE can be classified as: unidirectional and bidirectional PRE, and single-hop and multi-hop PRE, where the definitions are given in [8]. Many variants have been proposed to improve the flexibility and scalability of PRE and they focus on the single-hop bidirectional case [9]. These works include conditional PRE (CPRE) [10], identity-based PRE (IB-PRE) [11], [12], attribute-based PRE (AB-PRE) [13]. A hybrid PRE scheme (in general) was proposed [14] to unidirectional link ABE to IBE. This is the first cross-domain encryption scheme. The aforementioned AB-PRE schemes are weakly defined with CPA security. [15] achieved CCA security with any monotonic access structures and [16] further designed a functional PRE supporting deterministic finite automata. In fact, these AB-PRE schemes were built on top of attribute-based encryption [3], which was introduced to provide a more expressive way for data sharing [17], [18], [19]. However, previous cross-domain encryption schemes cannot bidirectionally bridge two types of encryption mechanisms. Moreover, they fail to support the search function in encryption switching.

Keyword Search. Song et al. proposed (symmetric) searchable encryption (SSE)to allow search over encrypted data using an encrypted keyword. Boneh et al. [5] seminally introduced the notion of *PEKS* to address the issues of the encrypted data sharing and complicated key management in SSE. Afterwards, combinable multi-keyword search schemes have been proposed to provide more expressiveness in search, such as public-key encryption scheme with conjunctive keyword search (PECKS), e.g, [20], [21], [22], and public key encryption with temporary keyword search (PETKS) [23]. To improve keyword privacy (i.e. adversary cannot guess what embedded into a given search trapdoor is), secure channel free-PEKS (SCF-PEKS) schemes [24], [25] and searchable encrypted keywords against insider attakcs (CR-IA) [25], [26] were proposed to resist outsider and insider attacks, respectively. Public key encryption with oblivious keyword search (PEOKS) [27] and public key encryption with authorized keyword search (PEAKS) [28], [29], [30] were proposed to permit authorized private search. However, these searchable encryption schemes cannot directly support the switching function of different mechanisms.

3 Preliminaries

3.1 System Model

We show the frame work of CDSS in Fig. 2, which consists of an IBE system, a PKE system and a switching entity.

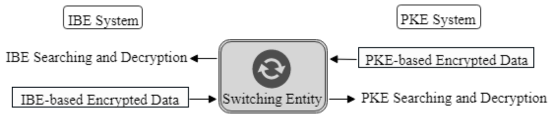


Fig. 2: CDSS Framework.

- *IBE System* deals with data encrypted under IBE format, where the IBE ciphertext is searchable.
- *PKE System* deals with data encrypted under PKE format, where the PKE ciphertext is searchable.
- *Switching Entity*, which is represented by a switching router, can bidirectionally execute the transformation of the ciphertexts belonging to different domains. That is, the switching entity can convert an IBE ciphertext form *(i.e. original domain)* into a PKE ciphertext form *(i.e. current domain)*, so that the switching ciphertext can be searched and decrypted with the PKE decryption algorithm; and vice versa.

3.2 Threat Model

This paper focuses on IBE and PKE systems, and regards their searchable cross-domain transformation as the main target. Assume that each plaintext is associated with a keyword (or a set of keyword) and is encrypted into the corresponding ciphertext via the encryption algorithm in original domain (either IBE or PKE).

Our system requires that some common parameters are pre-generated in the system initialization. The original domain and the current domain are built on these parameters and all associated operations including switching use the same public information. Our threat model makes the following assumptions. First, all of public keys and data streams transferred in the public channel are accessible. Second, each identity or keyword used in this paper is a short string which may suffer from brute-force guessing attacks. Third, we do not consider any collusion of entities from different domains. Finally, we assume the service provider to be a trust party who holds the master secret key, such that it can decrypt any IBE ciphertext.

We consider the cross-domain encryption switching that maintains retrieval function to execute keyword search before decryption. Assume that there exists an "honest-but-curious" adversary that follows specifications but attempts to compromise data and its associated keyword based on the following attacks: (i) The *chosen plaintext attack* models a case in which the adversary knows some of ciphertexts for arbitrary plaintexts (including original-domain ciphertext and current-domain ciphertext) and guesses the underlying plaintext. (ii) The *chosen keyword attack* models the case where an adversary knows some of the ciphertexts for arbitrary keywords (including original-domain ciphertext and current-domain ciphertext) and guesses the used keyword.

4 Algorithm Definition

We formally define CDSS as follows. The notations used in this paper is defined in Table 1.

TABLE 1: Notations.

pp	System public parameters.
M	Message.
w	Keyword.
msk, mpk	Master secret/public key.
ID	User identity.
sk_{IBE}, pk_{IBE}	IBE secret/public key.
sk_{PKE}, pk_{PKE}	PKE secret/public key.
$\mathcal{E}^{PKE}$	PKE ciphertext
$rk^{IBE\text{-}PKE}$	IBE-PKE switch key.
$\mathcal{E}^{IBE\text{-}PKE}$	IBE-PKE switch ciphertext
$rk^{PKE\text{-}IBE}$	PKE-IBE switch key.
$\mathcal{E}^{PKE\text{-}IBE}$	PKE-IBE switch ciphertext

Definition 1. *Under pp, a CDSS protocol includes the following algorithms.*

- Setup. Taking as input a security parameter λ, it outputs the master secret key msk and the master public key mpk. Assume mpk is implicitly included in the following algorithms.
- KeyGen. Taking as input a description δ and an alternative msk (it is necessary in IBE but not in PKE), it outputs a secret/public key pair (sk_δ, δ).
- Encrypt. Taking as input δ, the message $M \in \mathbb{G}_T$ and a keyword $w \in \{0,1\}^*$, it outputs the ciphertext $\mathcal{E}^\delta$.
- Switch_KeyGen$_{\delta-\xi}$. Taking as input either the tuple (sk_δ, ξ) or the tuple (sk_ξ,δ) and msk, it outputs a switch key $rk^{\delta-\xi}$, where δ and ξ are distinct descriptions.
- Switch. Taking as input the tuple ($rk^{\delta-\xi}$, $\mathcal{E}^\delta$), it outputs the ciphertext $\mathcal{E}^{\delta-\xi}$.
- Trapdoor. Taking as input sk_ξ and w, it outputs the trapdoor T^ξ.
- Test. Taking as input $\mathcal{E}^{\delta-\xi}$ and T^ξ, it outputs 1 if there is a match, and 0 otherwise.
- Decrypt. Taking as input $\mathcal{E}^{\delta-\xi}$ and sk_ξ, it outputs M, or $\perp$ indicating a fail decryption.

5 Formal Security Models

An adversary may follow specifications but try to compromise the underlying data and its associated keyword. The adversary can take some misbehaves, including the possible parameters accessing (the details can be found in the following queries). We also need two further assumptions. One is that the identity and the keyword may be guessed by the brute force attack, and the other is that no target secret information is available.

A CDSS protocol should guarantee security in two respects: the message secrecy and the keyword secrecy. For the former, we prevent an adversary, who is not given a valid secret key for decryption, from gaining access to the underlying message of both the original and the switched ciphertexts. For the latter, we need to guarantee that an adversary, without being given a valid trapdoor for search, cannot access the keyword from both the original and the switched ciphertexts. We build the security model with two games between an adversary $\mathcal{A}$ and a challenger $\mathcal{C}$, we call them the chosen plaintext attack (CPA) game and the chosen keyword attack (CKA) game.

5.1 Data Secrecy Model

Game 1. *A CDSS achieves the data secrecy (also known as CPA security) if satisfying that $\mathcal{A}$ cannot distinguish the message from the ciphertext and the switch ciphertext even though $\mathcal{A}$ is allowed to adaptively conduct some queries with restrictions.*

Setup. $\mathcal{C}$ gives pp, mpk to $\mathcal{A}$.
Phase 1. $\mathcal{A}$ issues the following queries.
PKE Key Query. $\mathcal{A}$ sends any PKE key query to $\mathcal{C}$. $\mathcal{C}$ runs the KeyGen$_{PKE}$ algorithm. $\mathcal{C}$ returns the public key pk_{PKE} for the target description, and returns the public/secret key pairs for any non-target descriptions.
IBE Key Query. $\mathcal{A}$ sends ID to $\mathcal{C}$. $\mathcal{C}$ runs the KeyGen$_{IBE}$ algorithm and returns the secret key sk_{IBE} and the public key pk_{IBE}.
Switch Key Query. $\mathcal{A}$ sends pk_δ, pk_ξ, w to $\mathcal{C}$, where pk_ξ is the target public key. $\mathcal{C}$ runs the Switch_KeyGen algorithm and returns the switch key $rk^{\delta-\xi}$, where we require that δ and ξ are from different domains. Namely, if δ represents IBE, while ξ represents PKE; vice versa.
Challenge. Once $\mathcal{A}$ decides that the Phase 1 is over, it submits two equal-length messages M_0, M_1, any keyword w and two description δ^*, ξ^*, where δ^*, ξ^* are restricted to be from different domains. $\mathcal{A}$ is restricted that he has only queried the switch key from any δ to the target ξ^*, and has queried neither the secret key for decryption for any description δ^* or ξ^*. $\mathcal{C}$ flips a coin $\epsilon \in \{0,1\}$ and returns the challenge ciphertext for (δ^*, M_ϵ) and the switch ciphertext for (ξ^*, M_ϵ).
Phase 2. As **Phase 1** with the condition established in challenge phase.
Guess. $\mathcal{A}$ outputs a guess ϵ' for ϵ.
$\mathcal{A}$ wins Game1 if $\epsilon' = \epsilon$.

5.2 Search Keyword Secrecy Model

Game 2. *A CDSS achieves the search keyword secrecy (also known as CKA security) if satisfying that $\mathcal{A}$ cannot distinguish one keyword from another even though $\mathcal{A}$ is allowed to adaptively conduct some queries with restrictions.*

Setup. As Game 1.
Phase 1. $\mathcal{A}$ is allowed to issue the following queries.
Key Query. $\mathcal{C}$ runs the KeyGen$_\delta$, KeyGen$_\xi$ algorithm and returns the public key for the target description. For non-target descriptions, $\mathcal{C}$ returns the public/secret key pairs.
Trapdoor Query. $\mathcal{A}$ sends w to $\mathcal{C}$. $\mathcal{C}$ runs the Trapdoor algorithm and returns the trapdoor T.
Switch Key Query. $\mathcal{A}$ sends pk_δ, pk_ξ, w to $\mathcal{C}$, where pk_ξ is the target public key. $\mathcal{C}$ runs the Swith_KeyGen algorithm and returns the switch key $rk^{\delta-\xi}$, where δ and ξ are also from different domains.
Challenge. Once $\mathcal{A}$ decides that the Phase 1 is over, it submits two equal-length keywords w_0, w_1, any message M and two target description δ^*, ξ^*. $\mathcal{A}$ is restricted that he has only queried the switch key from any δ to ξ^*, and has queried neither the secret key for δ^*, ξ^* nor the trapdoor for w_0, w_1 under the description δ^*, ξ^*. $\mathcal{C}$ flips a coin $\epsilon \in \{0,1\}$ and returns the searchable ciphertext for (δ^*, w_ϵ) and the switch searchable ciphertext for (ξ^*, w_ϵ).
Phase 2. As **Phase 1** with the condition established in challenge phase.
Guess. $\mathcal{A}$ outputs a guess ϵ' for ϵ.
$\mathcal{A}$ wins Game1 if $\epsilon' = \epsilon$.

6 The Proposed CDSS Protocol

6.1 Bilinear Pairing Operation [31]

Let $\mathbb{G}$ and $\mathbb{G}_T$ be two cyclic groups of the same prime order p. We say a map $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ is a symmetric bilinear pairing if

(1) for all $g \in \mathbb{G}$ and $a, b \in \mathbb{Z}_p$, we have $e\left(g^a, g^b\right) = e(g,g)^{ab}$.

(2) $e(g,g) \neq 1$ and it is efficient to compute $e(g,g)$ for all $g \in \mathbb{G}$.

A bilinear pairing group system $\mathcal{PG} = (\mathbb{G}, \mathbb{G}_T, e, p)$ is composed of the above defined objects.

6.2 Our Design

Our CDSS protocol is built from a pairing group system $\mathcal{PG}$. Let g, h be the generators of $\mathbb{G}$ and H_1, H_2 be cryptographic (collision resistant) hash functions, where $H_1 : \{0,1\}^* \to \mathbb{Z}_p$ and $H_2 : \{0,1\}^* \to \mathbb{G}$. We denote the public parameters as $pp = (g, h, H_1, H_2, \mathcal{PG})$. All of algorithms are built atop pp. We present the CDSS protocol in Fig. 3 and its details as follows.

Setup. The service provider initializes the switching system, where the service provider is trusted and manages both IBE mechanism and PKE mechanism. When used in the scenario described in Section 1, the service provider is Department of Health/Medicine. It selects $\alpha \in \mathbb{Z}_p$ and sets $msk = \alpha,\ mpk = g^{\alpha}$. pp and mpk are implicitly included in the following algorithms.

The service provider also shares a distinct secret value $z \in \mathbb{Z}_p$ with each sender.

KeyGen. This algorithm considers both IBE case and PKE case. This algorithm is run by both the service provider and the user in different encryption mechanisms.

- For IBE case, it outputs $d_{ID} = g^{\frac{1}{\alpha+H_1(ID)}}, h_{ID} = h^{\frac{1}{\alpha+H_1(ID)}}$. The user also chooses $\beta, u \in \mathbb{Z}_p$ and sets $sk_{IBE} = (d_{ID}, \beta, u), pk_{IBE} = (h^u, h^{\beta}, g^{\frac{\beta}{u}}, h^{\frac{u}{\beta}}, h_{ID})$.
- For PKE case, the user selects $x, y \in \mathbb{Z}_p$ and outputs $sk_{PKE} = (x, y), pk_{PKE} = (g^x, g^{xy}, g^{\frac{1}{x}}, g^y, g^{\frac{y}{x}}, h^y, h^{\frac{x}{y}})$.

Encrypt. This algorithm considers both IBE case and PKE case. This algorithm is run by the sender under different encryption mechanisms.

- For IBE case, it selects $r \in \mathbb{Z}_p$ and computes $\mathcal{E}^{IBE}$ as

$$\mathcal{E}^{IBE} = \begin{cases} X_1^I = g^{-r(\alpha+H_1(ID))}, \\ X_2^I = M \cdot e(g,g)^r, \\ X_3^I = h^{ru}, \\ X_4^I = e\left(h^{\beta}, H_2(w)\right)^r, \\ X_5^I = H_2(w)^{rz}. \end{cases}$$

- For PKE case, it selects $s \in \mathbb{Z}_p$ and computes $\mathcal{E}^{PKE}$ as

$$\mathcal{E}^{PKE} = \begin{cases} X_1^P = e(g,g)^s, \\ X_2^P = M \cdot e(g,g)^{xs}, \\ X_3^P = g^{-xys}, \\ X_4^P = g^{ys}, \\ X_5^P = h^{yzs}, \\ X_6^P = e\left(g^x, H_2(w)\right)^s, \\ X_7^P = H_2(w)^{sz}. \end{cases}$$

Switch_KeyGen. This algorithm considers both IBE-PKE case and PKE-IBE case. This algorithm is run by the service provider.

- For IBE-PKE case, it selects $t \in \mathbb{Z}_p$ and outputs a switch key $rk^{IBE-PKE}$ as

$$rk^{IBE-PKE} = \begin{cases} rk_1^{I\text{-}P} = g^{\frac{1}{x(\alpha+H_1(ID))}} h^t, \\ rk_2^{I\text{-}P} = (g^x)^{\frac{(\alpha+H_1(ID))t}{u}}, \\ rk_3^{I\text{-}P} = g^{\frac{ty}{x}}, \\ rk_4^{I\text{-}P} = e\left(g, H_2(w)\right)^t, \\ rk_5^{I\text{-}P} = h^{\frac{xu}{yz}}. \end{cases}$$

- For PKE-IBE case, it selects $t \in \mathbb{Z}_p$ and outputs a switch key $rk^{PKE-IBE}$ as

$$rk^{PKE-IBE} = \begin{cases} rk_1^{P\text{-}I} = g^{\frac{1}{y}} h^v, \\ rk_2^{P\text{-}I} = h^{xv} h_{ID}^z, \\ rk_3^{P\text{-}I} = h^{\frac{uv}{\beta}}, \\ rk_4^{P\text{-}I} = e\left(h, H_2(w)\right)^v, \\ rk_5^{P\text{-}I} = g^{\frac{\beta y}{uz}}. \end{cases}$$

Then the service provider can send the switch key to the switch party.

Switch. This algorithm considers both IBE-PKE case and PKE-IBE case. This algorithm is run by any switch party who has the switch key. Note that the switch party holds only the switch key but not msk, and hence cannot decrypt to obtain the message.

- For IBE-PKE case, it outputs the ciphertext $\mathcal{E}^{IBE-PKE}$ as

$$\mathcal{E}^{IBE-PKE} = \begin{cases} X_1^P = X_1^{I\text{-}P} = e\left(\left(X_1^I\right)^{-1}, rk_1^{I\text{-}P}\right), \\ X_2^P = X_2^{I\text{-}P} = X_2^I \cdot e\left(X_3^I, rk_2^{I\text{-}P}\right), \\ X_4^P = X_3^{I\text{-}P} = X_3^I \cdot rk_3^{I\text{-}P}, \\ X_6^P = X_4^{I\text{-}P} = e\left(X_5^I, rk_5^{I\text{-}P}\right) \cdot rk_4^{I\text{-}P}. \end{cases}$$

- For PKE-IBE case, it outputs the ciphertext $\mathcal{E}^{PKE-IBE}$ as

$$\mathcal{E}^{PKE-IBE} = \begin{cases} X_1^I = X_1^{P\text{-}I} = X_5^P, \\ X_2^I = X_2^{P\text{-}I} = X_2^P \cdot e\left(X_3^P, rk_1^{P\text{-}I}\right) \cdot \\ \qquad e\left(X_4^P, rk_2^{P\text{-}I}\right), \\ X_3^I = X_3^{P\text{-}I} = X_4^P \cdot rk_3^{P\text{-}I}, \\ X_4^I = X_4^{P\text{-}I} = e\left(X_7^P, rk_5^{P\text{-}I}\right) \cdot rk_4^{P\text{-}I}. \end{cases}$$

Trapdoor. This algorithm considers both IBE case and PKE case. This algorithm is run by the user in different encryption mechanisms.

- For IBE case, it outputs the trapdoor $T^{IBE} = H_2(w)^{\frac{\beta}{u}}$.
- For PKE case, it outputs the trapdoor $T^{PKE} = H_2(w)^{\frac{x}{y}}$.

Test. This algorithm considers both IBE case and PKE case. This algorithm is run by the server.

- For IBE case, it outputs 1 and returns the search result if $e\left(X_3^I, T^{IBE}\right) = X_4^I$.
- For PKE case, it outputs 1 and returns the search result if $e\left(X_4^P, T^{PKE}\right) = X_6^P$.

Decrypt. This algorithm considers both IBE case and PKE case. This algorithm is run by the user in different encryption mechanisms.

- For IBE case, it outputs $M = X_2^I \cdot e\left(X_1^I, d_{ID}\right)$.
- For PKE case, it outputs $M = \frac{X_2^P}{\left(X_1^P\right)^x}$.

Algorithm	**IBE**	**Switching**	**PKE**
Setup		Share pp, mpk.	
KeyGen	Compute d_{ID}, h_{ID}. Compute sk_{IBE}, pk_{IBE}.		Compute sk_{PKE}, pk_{PKE}.
Encrypt	Compute $\mathcal{E}^{IBE} = \left(X_1^I, X_2^I, \cdots, X_5^I\right)$,		Compute $\mathcal{E}^{PKE} = \left(X_1^P, X_2^P, \cdots, X_7^P\right)$,
Switch_KeyGen	**IBE→ PKE**	Compute $rk^{IBE\text{-}PKE} = \left(rk_1^{I\text{-}P}, rk_2^{I\text{-}P}, rk_3^{I\text{-}P}, rk_4^{I\text{-}P}, rk_5^{I\text{-}P}\right)$, $\xrightarrow{\qquad\qquad}$	
Switch		Compute $\mathcal{E}^{IBE\text{-}PKE} = \left(X_1^P, X_2^P, X_4^P, X_6^P\right)$ $\xrightarrow{\qquad\qquad}$	
Switch_KeyGen	**PKE←IBE**	Compute $rk^{PKE\text{-}IBE} = \left(rk_1^{P\text{-}I}, rk_2^{P\text{-}I}, rk_3^{P\text{-}I}, rk_4^{P\text{-}I}, rk_5^{P\text{-}I}\right)$, $\xleftarrow{\qquad\qquad}$	
Switch		Compute $\mathcal{E}^{PKE\text{-}IBE} = \left(X_1^I, X_2^I, X_3^I, X_4^I\right)$ $\xleftarrow{\qquad\qquad}$	
Trapdoor	Compute T^{IBE}		Compute T^{PKE}
Test	Check $e\left(X_3^I, T^{IBE}\right) = X_4^I$.		Check $e\left(X_4^P, T^{PKE}\right) = X_6^P$.
Decryption	Decrypt $M = X_2^I \cdot e\left(X_1^I, d_{ID}\right)$.		Decrypt $M = \frac{X_2^P}{\left(X_1^P\right)^x}$.

Fig. 3: Cross-Domain Encryption Switching with Keyword Search Protocol.

Correctness. We can verify the correctness as follows.

$$\begin{aligned}
e\left(X_3^I, T^{IBE}\right) &= e\left(g^{ys} \cdot h^{\frac{uv}{\beta}}, H_2(w)^{\frac{\beta}{u}}\right) \\
&= e\left(g, H_2(w)\right)^{\frac{ys\beta}{u}} e\left(h, H_2(w)\right)^v \\
&= e\left(X_7^P, rk_5^{P\text{-}I}\right) \cdot rk_4^{P\text{-}I} \\
&= X_4^I, \\
e\left(X_4^P, T^{PKE}\right) &= e\left(h^{ru} g^{\frac{ty}{x}}, H_2(w)^{\frac{x}{y}}\right) \\
&= e\left(X_5^I, rk_5^{I\text{-}P}\right) \cdot rk_4^{I\text{-}P} \\
&= X_6^P, \\
X_2^I \cdot e\left(X_1^I, d_{ID}\right) &= Me\left(g, g\right)^{xs} \cdot e\left(g^{-xys}, g^{\frac{1}{y}} h^v\right) \cdot \\
&\quad e\left(g^{ys}, h^{xv} h^{\frac{z}{\alpha + H_1(ID)}}\right) \cdot \\
&\quad e\left(h^{yzs}, g^{\frac{1}{\alpha + H_1(ID)}}\right) \\
&= M, \\
\frac{X_2^P}{\left(X_1^P\right)^x} &= \frac{M \cdot e(g,g)^r \cdot e\left(h^{ru}, (g^x)^{\frac{(\alpha + H_1(ID))t}{u}}\right)}{e\left(g^{r(\alpha + H_1(ID))}, g^{\frac{1}{x(\alpha + H_1(ID))}} h^t\right)^x} \\
&= M.
\end{aligned}$$

7 Security Analysis

We analyze the security of our CDSS in two directions, namely from IBE to PKE and from PKE to IBE, respectively. Before proceeding to the proof, we introduce the hard problems.

7.1 Hard Problems

The security of our proposed CDSS scheme is reduced to three hard assumptions, which are specific general decisional Diffie-Hellman exponent (GDDHE for short) assumptions introduced by Boneh, Boyen and Goh in [31]. We present them as follows.

Let g_0, P be the generator of $\mathbb{G}$ and $f(a), q(a)$ be two coprime polynomials with the degree deg $f(a) = n$ and deg $q(a) = 1$.

Problem 1.

$$\begin{aligned}
\text{Instance}: \quad & g_0^{f(a)q(a)}, g_0^{x^2 f(a)q(a)}, g_0^{x^2 y f(a)q(a)}, g_0^{x^2 a f(a)q(a)}, \\
& g_0^{xq(a)}, \cdots, g_0^{xa^{n-1}q(a)}, g_0^{xf(a)q(a)}, g_0^{xaf(a)q(a)}, \\
& g_0^{yq(a)}, \cdots, g_0^{ya^{n-1}q(a)}, g_0^{yf(a)}, g_0^{yf(a)q(a)}, \\
& g_0^{(1+kyz)f(a)}, g_0^{(1+kyz)q(a)}, \cdots, g_0^{(1+kyz)a^{n-1}q(a)}, \\
& e(g_0, P)^{xf(a)}, e(g_0, P)^{xq(a)}, \cdots, e(g_0, P)^{xa^{n-1}q(a)}, \\
& g_0^{ky^2 z f(a)q(a)}, g_0^{kxf(a)q(a)}, g_0^{kxzf(a)q(a)}, \\
& g_0^{xyf(a)q(a)}, g_0^{rkyzf(a)q(a)}, e(g_0, P)^{rkyzf(a)q(a)}, \\
& g_0^{kyzq(a)}, \cdots, g_0^{kyza^{n-1}q(a)}, g_0^{rxf(a)q^2(a)}, P^{rz}, T.
\end{aligned}$$

Output : Distinguish $T = e(g_0, g_0)^{rx^2 f^2(a) q^2(a)}$ or a random element in $\mathbb{G}_T$.

Problem 2.

$$\begin{aligned}
\text{Instance}: \quad & g_0^{xuy}, g_0^{x^2 uy}, g_0^{x^2 uy^2}, g_0^{uy}, g_0^{xuy^2}, g_0^{uy^2}, g_0^{\beta xy}, g_0^{x^2 y}, g_0^{bxuy}, \\
& g_0^{rxuy}, g_0^{rbxuyz}, g_0^{rxuyz}, h_0^{r\beta uyz}, e(g_0, g_0)^{rb\beta^2 xuy^2 z}, \\
& h_0^{\beta^2 yz}, h_0^{uyz}, h_0^{\beta y^2 z}, h_0^{\beta xz}, h_0^{\beta xu}, e(g_0, h_0)^{r\beta^2 xuy^2 z}, T
\end{aligned}$$

Output : Distinguish $T = e(g_0, h_0)^{rb\beta^2 xuy^2 z}$ or a random element in $\mathbb{G}_T$.

Problem 3.

Instance : $g_0^{xyzf(a)q(a)}, g_0^{xyzaf(a)q(a)}, g_0^{xy^2zf(a)q(a)}, g_0^{xyzq(a)}, \cdots, g_0^{xyza^{n-1}q(a)}, g_0^{x^2y^2zf(a)q(a)}, g_0^{x^2yzf(a)q(a)}, e(g_0,g_0)^{sx^2y^2z^2f^2(a)q^2(a)}, g_0^{yzf(a)q(a)}, g_0^{y^2zf(a)q(a)}, g_0^{xy^2f(a)q(a)}, g_0^{(1+k)xzf(a)q(a)}, g_0^{kyzf(a)+kx^2zf(a)q(a)}, e(g_0,P)^{kxyzf(a)q(a)}, e(g_0,P)^{sx^2yzf(a)q(a)}, g_0^{sxy^2zf(a)q(a)}, g_0^{sx^2y^2zf(a)q(a)}, g_0^{sky^2zf(a)q(a)}, g_0^{kxf(a)q(a)}, g_0^{ky^2f(a)q(a)}, g_0^{kxzf(a)q(a)}, g_0^{kyf(a)}, g_0^{kyq(a)}, \cdots, g_0^{kya^{n-1}q(a)}, P^{sz}, T$

Output : Distinguish $T = e(g_0,g_0)^{sx^3y^2z^2f^2(a)q^2(a)}$ or a random element in $\mathbb{G}_T$.

The generic complexity of the above hard problems can be covered by the analysis in [31] as it fits the general framework. Boneh, Boyen and Goh [31] also defined the hardness conditions and any problem fulfilling their conditions can be seen as one of GDDHE problems. It is quite clear that the proposed three hard problems satisfy the hardness condition, that is, the goal is independent of the given instance.

7.2 Data Secrecy of CDSS from IBE to PKE

When switching from IBE to PKE, the decryption needs the secret key x. The adversary cannot obtain x even if he/she knows switch keys. This means that it is possible for the adversary to distinguish the message from the ciphertext or the switch ciphertext. Hence, CPA security holds in CDSS protocol from IBE to PKE.

Theorem 1. *The CDSS protocol from IBE to PKE achieves the CPA security under the Game 1 if the* Problem 1 *is hard.*

Proof. Suppose an adversary $\mathcal{A}$ can break our protocol from IBE to PKE in Game 1 with a non-negligible advantage, there exists an simulator $\mathcal{B}$, who receives Problem 1 instances and attempts to use $\mathcal{A}$ to solve the *Problem 1* with a non-negligible advantage.

Setup. $\mathcal{B}$ runs the Setup algorithm and publishes the system parameters. $\mathcal{B}$ implicitly sets $\alpha = a$, $g = g_0^{xf(a)q(a)}$, $h = g_0^{kyzf(a)q(a)}$, where

$$f(a) = (a+a_1)\cdots(a+a_n), q(a) = (a+a^*).$$

We also define $f_i(a) = \frac{f(a)}{a+a_i}$. $\mathcal{B}$ computes $g^\alpha = g_0^{xaf(a)q(a)}$. Then $\mathcal{B}$ issues the public parameter $pp = (g,h)$ and the master public key $mpk = g^\alpha$ to $\mathcal{A}$.

H_1-Query. $\mathcal{B}$ maintains a H_1 list $L(ID_i, c_i, a_i, h_i)$, which are initially empty. Upon receiving an H_1 query for ID_i, if ID_i is in the list L, $\mathcal{B}$ returns the corresponding h_i to $\mathcal{A}$. Otherwise, $\mathcal{B}$ picks a bit $c_i \in \{0,1\}$ and sets the hash value h_i as follows.

$$h_i = H_1(ID_i) = \begin{cases} a^*, & \text{if } c_i = 0, \\ a_i, & \text{if } c_i = 1. \end{cases}$$

Then $\mathcal{B}$ adds (ID_i, c_i, a_i, h_i) to the list and returns h_i to $\mathcal{A}$.

H_2-Query. $\mathcal{B}$ maintains a H_2 list $L'(w_i, c'_i, b_i, h'_i)$, which are initially empty. Upon receiving an H_2 query for w_i, if w_i is in the list L', $\mathcal{B}$ returns the corresponding h'_i to $\mathcal{A}$. Otherwise, $\mathcal{B}$ picks a bit $c'_i \in \{0,1\}, b_i \in \mathbb{Z}_p$ and sets the hash value $h'_i = H_2(w_i) = P^{b_i}$. Then $\mathcal{B}$ adds (w_i, c'_i, b_i, h'_i) to the list and returns h'_i to $\mathcal{A}$.

Phase 1. $\mathcal{A}$ can make queries as follows.

PKE Key Query. For target public key $pk_{PKE} = \left(g^x, g^{xy}, g^{\frac{1}{x}}, g^y, g^{\frac{y}{x}}, h^y, h^{\frac{x}{y}}\right)$, $\mathcal{B}$ can compute

$$g^x = g_0^{x^2f(a)q(a)}, g^{xy} = g_0^{x^2yf(a)q(a)}, g^{\frac{1}{x}} = g_0^{f(a)q(a)},$$
$$g^y = g_0^{xyf(a)q(a)}, g^{\frac{y}{x}} = g_0^{yf(a)q(a)},$$
$$h^y = g_0^{ky^2zf(a)q(a)}, h^{\frac{x}{y}} = g_0^{kxzf(a)q(a)},$$

which are computable from the instance. Then $\mathcal{B}$ sends pk_{PKE} to $\mathcal{A}$. For non-target PKE key, $\mathcal{B}$ runs the KeyGen_{PKE} algorithm to generate any public/secret key pair.

IBE Key Query. $\mathcal{A}$ sends ID_i to $\mathcal{B}$ for the IBE secret key query. $\mathcal{B}$ runs the KeyGen_{IBE} algorithm and returns the secret key and public key. Let (ID_i, c_i, a_i, h_i) be the corresponding tuple on the L list.

For the secret key query,

- If $c_i = 0$, abort.
- If $c_i = 1$, we have $h_i = H_1(ID_i) = a_i$. $\mathcal{B}$ responds the IBE secret key for ID_i with $d_{ID_i} = g_0^{xf_i(a)q(a)}$, where
$$d_{ID_i} = g^{\frac{1}{\alpha+H_1(ID_i)}} = g_0^{\frac{xf(a)q(a)}{\alpha+a_i}} = g_0^{xf_i(a)q(a)}.$$
d_{ID_i} is computable from the elements $g_0^{xq(a)}, g_0^{xaq(a)}, \cdots, g_0^{xa^{n-1}q(a)}$ in the instance,

$\mathcal{B}$ randomly selects $\beta, u \in \mathbb{Z}_p$ and returns $sk_{IBE} = \{d_{ID_i}, \beta, u\}$ to $\mathcal{A}$.

For the public key query,

- If $c_i = 0$, we have $h_i = H_1(ID_i) = a^*$. $\mathcal{B}$ responds the IBE public key for ID_i with $h^{\frac{1}{\alpha+H_1(ID_i)}} = g_0^{kyzf(a)}$.
- If $c_i = 1$, we have $h_i = H_1(ID_i) = a_i$. $\mathcal{B}$ responds the IBE public key for ID_i with $h^{\frac{1}{\alpha+H_1(ID_i)}} = g_0^{kyzf_i(a)q(a)}$, which is computable from the elements $g_0^{kyzq(a)}, g_0^{kyzaq(a)}, \cdots, g_0^{kyza^{n-1}q(a)}$ in the instance,

$\mathcal{B}$ also computes $h^u, h^\beta, g^{\frac{\beta}{u}}, h^{\frac{u}{\beta}}$ and returns $pk_{IBE} = \{h^u, h^\beta, g^{\frac{\beta}{u}}, h^{\frac{u}{\beta}}, h^{\frac{1}{\alpha+H_1(ID_i)}}\}$ to $\mathcal{A}$.

Switch Key Query. $\mathcal{A}$ sends the target PKE public key pk_{PKE}, the identity ID_i and the keyword w_i to $\mathcal{B}$. $\mathcal{B}$ runs the Switch_KeyGen algorithm and responds the switch key $rk^{IBE-PKE}$. Let (ID_i, c_i, a_i, h_i) be the corresponding tuple on the L list and (w_i, c'_i, b_i, h'_i) be the corresponding tuple on the L' list.

- If $c_i = 0$, we have $h_i = H_1(ID_i) = a^*, h'_i = H_2(w_i) = P^{b_i}$. $\mathcal{B}$ chooses $t' \in \mathbb{Z}_p$ and responds the switch key as

$$rk_1^{I\text{-}P} = g_0^{(1+kyz)f(a)} \cdot h^{t'},$$
$$rk_2^{I\text{-}P} = g_0^{\frac{x^2f(a)q(a)}{u}} \cdot g_0^{\frac{x^2f(a)q(a)(a+a^*)t'}{u}},$$
$$rk_3^{I\text{-}P} = g_0^{yf(a)} \cdot g^{\frac{yt'}{x}},$$
$$rk_4^{I\text{-}P} = e\left(g_0, P\right)^{xf(a)b_i} \cdot e\left(g, P\right)^{t'b_i},$$
$$rk_5^{I\text{-}P} = g_0^{kxf(a)q(a)u}$$

We implicitly set $t = \frac{1}{q(a)} + t'$ and verify it as

$$\begin{aligned} rk_1^{I\text{-}P} &= g^{\frac{1}{x(\alpha+H_1(ID_i))}} h^t \\ &= g_0^{(1+kyz)f(a)} \cdot h^{t'}, \\ rk_2^{I\text{-}P} &= g^{\frac{x(\alpha+H_1(ID_i))t}{u}} \\ &= g_0^{\frac{x^2 f(a)q(a)}{u}} \cdot g_0^{\frac{x^2 f(a)q(a)(a+a^*)t'}{u}}, \\ rk_3^{I\text{-}P} &= g^{\frac{ty}{x}} = g_0^{yf(a)} \cdot g^{\frac{yt'}{x}}, \\ rk_4^{I\text{-}P} &= e(g, H_2(w_i))^t \\ &= e\left(g_0, P\right)^{xf(a)b_i} \cdot e\left(g, P\right)^{t'b_i}, \\ rk_5^{I\text{-}P} &= g^{\frac{\beta y}{uz}} = g_0^{kxf(a)q(a)u} \end{aligned}$$

- If $c_i = 1$, we have $h_i = H_1(ID_i) = a_i, h'_i = H_2(w_i) = P^{b_i}$. $\mathcal{B}$ chooses $t' \in \mathbb{Z}_p$ and responds the switch key as

$$\begin{aligned} rk_1^{I\text{-}P} &= g_0^{(1+kyz)f_i(a)q(a)} \cdot h^{t'}, \\ rk_2^{I\text{-}P} &= g_0^{\frac{x^2 f(a)q(a)}{u}} \cdot g_0^{\frac{x^2 f(a)q(a)(a+a_i)t'}{u}}, \\ rk_3^{I\text{-}P} &= g_0^{yf_i(a)q(a)} \cdot g^{\frac{yt'}{x}}, \\ rk_4^{I\text{-}P} &= e\left(g_0, P\right)^{xf_i(a)q(a)b_i} \cdot e\left(g, P\right)^{t'b_i}, \\ rk_5^{I\text{-}P} &= g_0^{kxf(a)q(a)u} \end{aligned}$$

We implicitly set $t = \frac{1}{a+a_i} + t'$ and verify it as

$$\begin{aligned} rk_1^{I\text{-}P} &= g^{\frac{1}{x(\alpha+H_1(ID_i))}} h^t \\ &= g_0^{(1+kyz)f_i(a)q(a)} \cdot h^{t'}, \\ rk_2^{I\text{-}P} &= g^{\frac{x(\alpha+H_1(ID_i))t}{u}} \\ &= g_0^{\frac{x^2 f(a)q(a)}{u}} \cdot g_0^{\frac{x^2 f(a)q(a)(a+a_i)t'}{u}}, \\ rk_3^{I\text{-}P} &= g^{\frac{ty}{x}} = g_0^{yf_i(a)q(a)} \cdot g^{\frac{yt'}{x}}, \\ rk_4^{I\text{-}P} &= e(g, H_2(w_i))^t \\ &= e\left(g_0, P\right)^{xf_i(a)q(a)b_i} \cdot e\left(g, P\right)^{t'b_i}, \\ rk_5^{I\text{-}P} &= h^{\frac{xu}{yz}} = g_0^{kxf(a)q(a)u}. \end{aligned}$$

Challenge. $\mathcal{A}$ outputs ID^*, two messages M_0, M_1 and a keyword w^* on which it wishes to be challenged. Here we restrict that $\mathcal{A}$ is only allowed to query the switch key for the target PKE public key pk_{PKE}. Let (ID^*, c^*, a^*, h_*) be the corresponding tuple on the L list and (w^*, c'^*, b^*, h'_*) be the corresponding tuple on the L' list. we have $h'_* = H_2(w^*) = P^{b^*}$ $\mathcal{B}$ select $\epsilon \in \{0, 1\}$ and responds the challenge ciphertext to $\mathcal{A}$ as follows.

- If $c^* = 1$, abort.
- If $c^* = 0$, we have $h_* = H_1(ID^*) = a^*$. $\mathcal{B}$ responds the challenge ciphertext $\mathcal{E}^{IBE}$ as

$$X_1^I = g_0^{-rxf(a)q^2(a)}, X_2^I = M_\epsilon \cdot T, X_3^I = g_0^{rkyzf(a)q(a)u}, \\ X_4^I = e(g_0, P)^{rkyzf(a)q(a)\beta b^*}, X_5^I = P^{rzb^*},$$

If $T = e\left(g_0, g_0\right)^{rx^2 f^2(a)q^2(a)}$, one can verify the correctness of the above ciphertext by

$$\begin{aligned} X_1^I &= g^{-r(\alpha+H_1(ID^*))} = g_0^{-rxf(a)q^2(a)}, \\ X_2^I &= M_\epsilon \cdot e\left(g, g\right)^r \\ &= M_\epsilon \cdot e\left(g_0, g_0\right)^{rx^2 f^2(a)q^2(a)} \\ &= M_\epsilon \cdot T, \\ X_3^I &= h^{ru} = g_0^{rkyzf(a)q(a)u}, \\ X_4^I &= e(h^\beta, H_2(w^*))^r \\ &= e(g_0, P)^{rkyzf(a)q(a)\beta b^*}, \\ X_5^I &= H_2(w^*)^{rz} = P^{rzb^*}, \end{aligned}$$

Therefore, $\mathcal{E}^{IBE}$ is a valid challenge ciphertext.

If T is a random element in $\mathbb{G}_T$, the challenge ciphertext will be random from the $\mathcal{A}$'s view.

Then $\mathcal{A}$ can generate the switch ciphertext $\mathcal{E}^{IBE-PKE}$ with the received challenge ciphertext and the switch key.

Phase 2. As **Phase 1** with the condition established in challenge phase.

Guess. Eventually $\mathcal{A}$ outputs a guess $\epsilon' \in \{0, 1\}$ for ϵ. □

7.3 Search Keyword Secrecy of CDSS from IBE to PKE

When switching from IBE to PKE, the test needs the trapdoor $T^{PKE} = H_2(w)^{\frac{x}{y}}$. The adversary cannot obtain T^{PKE} even if he/she can queries other secret keys or trapdoors. This means that it is possible for the adversary to guess the keyword from the ciphertext or switch ciphertext. Hence, CKA security holds in CDSS protocol from IBE to PKE.

Theorem 2. *The CDSS protocol from IBE to PKE achieves the CKA security under the Game 2 if the Problem 2 is hard.*

Proof. In this proof, we assume that the adversary cannot be a valid sender who has got the secret value z. Then if an adversary $\mathcal{A}$ can break our protocol from IBE to PKE in Game 2 with a non-negligible advantage, there exists an simulator $\mathcal{B}$, who receives Problem 2 instances and can solve the *Problem 2* with a non-negligible advantage.

Setup. $\mathcal{B}$ runs the Setup algorithm and publishes the system parameters. $\mathcal{B}$ implicitly sets $g = g_0^{xuy}, h = h_0^{\beta yz}$ and chooses $\alpha \in \mathbb{Z}_p$. Then $\mathcal{B}$ issues the public parameter $pp = (g, h)$ and the master public key $mpk = g^\alpha$ to $\mathcal{A}$.

H_2**-Query**. $\mathcal{B}$ maintains a H_2 list $L'(w_i, c'_i, b_i, h'_i)$, which are initially empty. Upon receiving an H_2 query for w_i, if w_i is in the list L', $\mathcal{B}$ returns the corresponding h'_i to $\mathcal{A}$. Otherwise, $\mathcal{B}$ picks a bit $c'_i \in \{0, 1\}$ and sets the hash value h'_i as follows.

$$h'_i = H_2(w_i) = \begin{cases} g^b g^{b_i}, & \text{if } c'_i = 0, \\ g^{b_i}, & \text{if } c'_i = 1. \end{cases}$$

Then $\mathcal{B}$ adds (w_i, c'_i, b_i, h'_i) to the list and returns h'_i to $\mathcal{A}$.

Phase 1. $\mathcal{A}$ can make queries as follows.

Key Query. For target public key

$$pk_{PKE} = \left(g^x, g^{xy}, g^{\frac{1}{x}}, g^y, g^{\frac{y}{x}}, h^y, h^{\frac{x}{y}}\right),$$

$$pk_{IBE} = \left(h^u, h^\beta, g^{\frac{\beta}{u}}, h^{\frac{u}{\beta}}\right),$$

$\mathcal{B}$ computes $g^x = g_0^{x^2uy}, g^{xy} = g_0^{x^2uy^2}, g^{\frac{1}{x}} = g_0^{uy}, g^y = g_0^x, g^{\frac{y}{x}} = g_0^{uy^2}, h^y = h_0^{\beta y^2 z}, h^{\frac{x}{y}} = h_0^{\beta xz}$ and $h^u = h_0^{u\beta yz}, h^\beta = h_0^{\beta^2 yz}, g^{\frac{\beta}{u}} = g_0^{\beta xy}, h^{\frac{u}{\beta}} = h_0^{uyz}$, which can be from the instance.

For non-target public key, $\mathcal{B}$ runs the $\mathsf{KeyGen}_{PKE}, \mathsf{KeyGen}_{IBE}$ algorithms to generate public/secret key pairs, which are independent of the target one. Then $\mathcal{B}$ sends the public key to $\mathcal{A}$.

Trapdoor Query. $\mathcal{A}$ sends w_i to $\mathcal{B}$ for the trapdoor query. $\mathcal{B}$ runs the Trapdoor algorithm and returns the trapdoor T. Let (w_i, c_i, a_i, h_i) be the corresponding tuple on the L' list.

- If $c_i = 0$, abort.
- If $c_i = 1$, we have $h'_i = H_2(w_i) = g^{b_i}$. $\mathcal{B}$ responds the trapdoor for w_i with $T^{IBE} = g_0^{\beta xyb_i}, T^{PKE} =$

$g_0^{x^2ub_i}$, where $T^{IBE} = H_2(w_i)^{\frac{\beta}{u}} = g_0^{b^2xza_i}, T^{PKE} = H_2(w_i)^{\frac{x}{y}} = g_0^{x^2ub_i}$.

Both trapdoors can be obtained from the elements in the instance,

Switch Key Query. $\mathcal{A}$ sends pk_{PKE}, pk_{IBE}, w_i to $\mathcal{B}$. $\mathcal{B}$ runs the Switch_KeyGen algorithm and responds the switch key $rk^{IBE-PKE}$. Let (w_i, c_i', b_i, h_i') be the corresponding tuple on the L' list. $\mathcal{B}$ chooses $\alpha, t \in \mathbb{Z}_p$ and responds the switch key as

$$rk_1^{I\text{-}P} = \left(g^{\frac{1}{x}}\right)^{\frac{1}{(\alpha+H_1(ID))}} h^t, rk_2^{I\text{-}P} = \left(g_0^{x^2y}\right)^{(\alpha+H_1(ID))t},$$
$$rk_3^{I\text{-}P} = \left(g^{\frac{y}{x}}\right)^t, rk_4^{I\text{-}P} = e\left(g, h_i'\right)^t, rk_5^{I\text{-}P} = h_0^{\beta xy}.$$

Challenge. $\mathcal{A}$ outputs pk_{IBE}, pk_{PKE} and two same-length keywords w_0, w_1 on which it wishes to be challenged. Here we restrict that $\mathcal{A}$ is only allowed to query the switch key for the target PKE public key pk_{PKE}. Let (w_i, c_i', b_i, h_i') be the corresponding tuple on the L' list. $\mathcal{B}$ select $\epsilon \in \{0,1\}$ and responds the challenge searchable ciphertext to $\mathcal{A}$ as follows.

- If $c^* = 1$, abort.
- If $c^* = 0$, we have $h_\epsilon' = H_2(w_\epsilon) = g^b g^{b_\epsilon}$. $\mathcal{B}$ responds the challenge searchable ciphertext $\mathcal{E}^{IBE}$ as

$$\begin{aligned} X_1^I = &\ (g_0^{rxuy})^{\alpha+H_1(ID)}, X_2^I = Me\left(g_0, g_0\right)^{rx^2u^2y^2}, \\ X_3^I = &\ h_0^{r\beta uyz}, X_4^I = T \cdot e\left(g_0, h_0\right)^{r\beta^2xuy^2zb_\epsilon}, \\ X_5^I = &\ g_0^{rbxuyz} g_0^{rxuyzb_\epsilon}, \end{aligned}$$

If $T = e\left(g_0, h_0\right)^{rb\beta^2xuy^2z}$, one can verify the correctness as

$$\begin{aligned} X_1^I &= g^{-r(\alpha+H_1(ID))} = (g_0^{rxuy})^{\alpha+H_1(ID)}, \\ X_2^I &= M \cdot e\left(g, g\right)^r = Me\left(g_0, g_0\right)^{rx^2u^2y^2}, \\ X_3^I &= h^{ru} = h_0^{r\beta uyz}, \\ X_4^I &= e(h^\beta, H_2(w^*))^r \\ &= e\left(h_0^{\beta^2yz}, g_0^{bxuy} g_0^{xuyb_\epsilon}\right)^r \\ &= T \cdot e\left(g_0, h_0\right)^{r\beta^2xuy^2zb_\epsilon}, \\ X_5^I &= H_2(w^*)^{rz} = \left(g^b g^{b_\epsilon}\right)^r = g_0^{rbxuyz} g_0^{rxuyzb_\epsilon}. \end{aligned}$$

Therefore, $\mathcal{E}^{IBE}$ is a valid challenge ciphertext.

If T is a random element in $\mathbb{G}_T$, the challenge ciphertext will be random from the $\mathcal{A}$'s view.

Then $\mathcal{A}$ can generate the switch ciphertext $\mathcal{E}^{IBE-PKE}$ with the received challenge ciphertext and the switch key.

Phase 2. As **Phase 1** with the condition established in challenge phase.

Guess. Eventually $\mathcal{A}$ outputs a guess $\epsilon' \in \{0,1\}$ for ϵ. □

7.4 Data and Search Keyword Secrecy from PKE to IBE

When switching from PKE to IBE, the decryption needs the secret key $d_{ID} = g^{\frac{1}{\alpha+H_1(ID)}}$. The adversary cannot obtain d_{ID} even if he/she can query other secret keys or switch keys due to the exponent-inversion structure of d_{ID}. This means that it is possible for the adversary to distinguish the message from the ciphertext or the switch ciphertext. Hence, CPA security holds in CDSS protocol from PKE to IBE.

Theorem 3. *The CDSS from PKE to IBE achieves the CPA security under the Game 1 if the Problem 3 is hard.*

When switching from PKE to IBE, the test needs the trapdoor $T^{IBE} = H_2(w)^{\frac{\beta}{u}}$. The adversary cannot obtain T^{IBE} even if he/she can query other secret keys trapdoors. This means that it is possible for the adversary to guess the keyword from the ciphertext or switch ciphertext. Hence, CKA security holds in CDSS protocol from PKE to IBE.

Theorem 4. *The CDSS protocol from PKE to IBE achieves the CKA security under the Game 2 if the Problem 2 is hard.*

The proofs are similar to Theorem 1 and 2.

8 Efficiency Analysis

We only present the efficiency analysis for our CDSS protocol due to no comparable protocols in the literature, where the efficiency is analyzed from both theoretical and practical levels. We consider the computational and communication costs for a trusted service provider, a system user, and a server. To better show the theoretical computational/communication cost ratio accordingly, we use the four tables (TABLE 2-TABLE 5) to illustrate different phases, different entities and different channels in a IBE→PKE/PKE→IBE switching system.

To present a fair computational analysis, we denote an exponent cost in $\mathbb{G}$ and a pairing cost in $\mathbb{G}_T$ as exp and $pair$, respectively. TABLE 2 lists the analysis when switching from IBE to PKE. From TABLE 2, we can see that most of the pairing costs are offloaded to the server and the switch party. In terms of pairing operation, a user needs to take 3 pairings cost of which one is for the decryption and two is for encryption. When switching from PKE to IBE, a user costs only 2 pairings, less than 50% of the total pairings cost and meanwhile, there is no extra cost for the service provider in KeyGen algorithm. The detailed analysis is presented in TABLE 3. In both tables, we also show the total computational cost in a PKE-IBE/IBE-PKE switching system, which is denoted as "Total".

In the theoretical communication analysis, we use mpk, msk, pk, sk, CT, T, rk to denote master secret key, master public key, public key, secret key, ciphertext, search token and switch token, respectively. We further use $|\mathbb{Z}_p|$, $|\mathbb{G}|$ and $|\mathbb{G}_T|$ to denote the size of an element in groups $\mathbb{Z}_p$, $\mathbb{G}$ and $\mathbb{G}_T$, respectively. We show the analysis in TABLE 4 and TABLE 5 in terms of the IBE → PKE switching and the PKE → IBE switching. Similar to the computational cost analysis, we also list total communication cost in a PKE-IBE/IBE-PKE switching system. From a system user's point of view, the communication complexity in switching and search are significantly cost-effective w.r.t. both of the tables, in which the user only needs to send an element in $\mathbb{G}$ to the server to fulfill a search, while it can request $4|\mathbb{G}| + |\mathbb{G}_T|$ communication cost in switch. The most expensive part relies on ciphertext, taking $2|\mathbb{G}_T|$ and $3|\mathbb{G}_T|$ in IBE → PKE and PKE → IBE, respectively.

We implement our CDSS on a Mac pro with 2.2GHz Intel Core i7 CPU and 16GB 1600 MHz DDR3 memory. In our implementation, we use PBC library[1], which is one

1. https://crypto.stanford.edu/pbc/

TABLE 2: Theoretical IBE $\rightarrow$ PKE Computational Cost

	IBE $\rightarrow$ PKE Computational Cost							
Cost	Setup	KeyGen	Enc	Switch_KeyGen	Switch	Trapdoor	Test	Dec
exp	1	6	6	6	0	1	0	0
$pair$	0	0	2	1	3	0	1	1
Service provider (Setup)	$1exp$							
Service provider (KeyGen)	$6exp$							
Server (Test)	$1pair$							
Service provider (Switch_keygen)	$1pair$							
Switch party (Switch)	$3pair$							
User	$13exp + 3pair$							
Total	$20exp + 8pair$							

TABLE 3: Theoretical PKE $\rightarrow$ IBE Computational Cost

	PKE $\rightarrow$ IBE Computational Cost							
Cost	Setup	KeyGen	Enc	Switch_KeyGen	Switch	Trapdoor	Test	Dec
exp	1	7	8	7	0	1	0	1
$pair$	0	0	3	1	3	0	1	0
Servise provider (Setup)	$1exp$							
Servise provider (KeyGen)	0							
Server (test)	$1pair$							
Service provider (Switch_keygen)	$1pair$							
Switch party (Switch)	$3pair$							
User	$24exp + 2pair$							
Total	$25exp + 7pair$							

TABLE 4: Theoretical IBE $\rightarrow$ PKE Communication Cost

	IBE $\rightarrow$ PKE Communication Cost						
Groups	mpk	msk	pk	sk	T	rk	CT
$\mathbb{Z}_q$	0	1	0	2	0	0	0
$\mathbb{G}$	1	0	5	1	1	4	3
$\mathbb{G}_T$	0	0	0	0	0	1	2
Service provider $\rightarrow$ User (KeyGen)	$2\vert\mathbb{Z}_p\vert + 6\vert\mathbb{G}\vert$						
User $\rightarrow$ Server (Enc)	$3\vert\mathbb{G}\vert + 2\vert\mathbb{G}_T\vert$						
Service provider $\rightarrow$ Switch party (Switch)	$4\vert\mathbb{G}\vert + \vert\mathbb{G}_T\vert$						
User $\rightarrow$ Server (Search)	$\vert\mathbb{G}\vert$						
Total	$2\vert\mathbb{Z}_p\vert + 14\vert\mathbb{G}\vert + 3\vert\mathbb{G}_T\vert$						

TABLE 5: Theoretical PKE $\rightarrow$ IBE Communication Cost

	PKE $\rightarrow$ IBE Communication Cost						
Groups	mpk	msk	pk	sk	T	rk	CT
$\mathbb{Z}_q$	0	1	0	2	0	0	0
$\mathbb{G}$	1	0	7	0	1	4	4
$\mathbb{G}_T$	0	0	0	0	0	1	3
Service provider $\rightarrow$ User (KeyGen)	0						
User $\rightarrow$ Server (Enc)	$4\vert\mathbb{G}\vert + 3\vert\mathbb{G}_T\vert$						
Service provider $\rightarrow$ Switch party (Switch)	$4\vert\mathbb{G}\vert + \vert\mathbb{G}_T\vert$						
User $\rightarrow$ Server (Search)	$\vert\mathbb{G}\vert$						
Total	$9\vert\mathbb{G}\vert + 4\vert\mathbb{G}_T\vert$						

of the most widely used library for pairing operation. We choose the symmetric pairing which is constructed based on the curve $y^2 = x^3 + x$ over the finite field F_p for some prime number p satsifying $p = 3 \bmod 4$, and both G_1 and G_2 are the group of points on $E(F_p)$. Table 6 shows the computational cost for each of the operations in our system.

The experiment shows the scalability of our system regarding the number of users and files during the process. For the simplicity of the demonstration, we apply the one keyword search scenario with multiple files and multiple users to test our computational cost and the communication cost. For the computational cost, we perform the encryption, decryption and the corresponding switching operations for 20000 independent files. And we categorize the cost for the IBE, PKE and the Switching systems accordingly. Figure 4 shows that the switching entity is responsible for most of the computational cost compared with the IBE and PKE calculation. When the number of the files reaches 20000, IBE, PKE and Switching entity systems will take around 300, 300 and 850 seconds to finish the process.

For the communication cost, we consider different transfer channels, i.e., Service provider $\rightarrow$ user, Service provider$\rightarrow$ Switch party and User$\rightarrow$ server. Fig. 5 shows the communication cost from the service provider to user,

TABLE 6: Experimental Computational Cost for each operation

	KeyGen	Encrypt	IBE→ PKE Switch-KeyGen	IBE→ PKE Switch	PKE→ IBE Switch-KeyGen	PKE→ IBE Switch	Trapdoor	Test	Encryption
IBE	0.007792s	0.011997s	-	-	-	-	0.001959s	0.004679s	0.006420*s*s
Switching	-	-	0.010252s	0.002614s	0.010879s	0.018809s -	-	-	
PKE	0.010879s	0.009694s	-	-	-	-	0.001959s	0.004936s	0.003444s

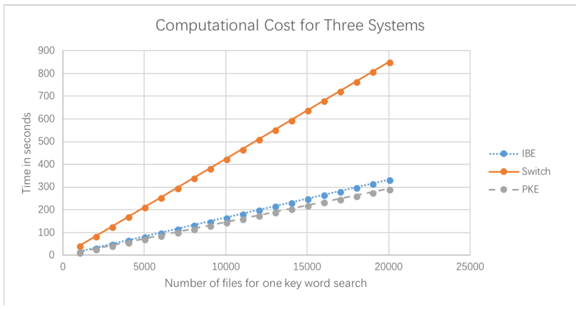


Fig. 4: Computational cost for three systems

where the number of users is up to 6000. Fig. 6 shows the communication cost from the service provider to switch party, where the number of ciphertexts is up to 6000. Fig. 7 shows the communication cost from user to server, where the number of ciphertexts is also up to 6000.

As one would expect that as the number of users and files grow large, the corresponding computational and communication cost grow in a linear pattern, which demonstrates the scalability of our system. While we demonstrate the single thread scenario, multiply users would query the service at the same time. For example in our implemented system, if 3000 users simultaneously asked for the switching service on 5000 files in the IBE to PKE scenario, then around 2M data will need to be transmitted to the switching entity, and the switching entity will take around 3 minutes in average to process all the requests. Thus as the service requirement grows, switching entity will be become a bottleneck in performance. In the future, we can consider to build hierarchical switching entity to disperse the workload.

Acknowledgements

C. Dong is supported by an EPSRC grant (Grant No. EP/M013561/2). It is also supported in part by the National Research Foundation, Prime Ministers Office, Singapore, under its Corporate Laboratory@University Scheme, National University of Singapore, and Singapore Telecommunications Ltd., in part by National Natural Science Foundation of China under Grant No. 61702212 and the research funds of CCNU from colleges basic research and operation of MOE under Grand No. CCNU16A05040.

References

[1] M. Blaze, G. Bleumer, and M. Strauss, “Divertible protocols and atomic proxy cryptography,” in *EUROCRYPT '98*, ser. LNCS, vol. 1403, 1998, pp. 127–144.
[2] R. Canetti and S. Hohenberger, “Chosen-ciphertext secure proxy re-encryption,” in *CCS 2007*, 2007, pp. 185–194.
[3] V. Goyal, O. Pandey, A. Sahai, and B. Waters, “Attribute-based encryption for fine-grained access control of encrypted data,” in *Proceedings of the 13th ACM conference on Computer and communications security*. Acm, 2006, pp. 89–98.
[4] J. Weng, R. H. Deng, S. Liu, and K. Chen, “Chosen-ciphertext secure bidirectional proxy re-encryption schemes without pairings,” *Inf. Sci.*, vol. 180, no. 24, pp. 5077–5089, 2010.
[5] D. Boneh, G. D. Crescenzo, R. Ostrovsky, and G. Persiano, “Public key encryption with keyword search,” in *EUROCRYPT 2004*, ser. LNCS, vol. 3027, 2004, pp. 506–522.
[6] G. Couteau, T. Peters, and D. Pointcheval, “Encryption switching protocols,” in *CRYPTO 2016*, ser. LNCS, vol. 9814, 2016, pp. 308–338.
[7] G. Castagnos, L. Imbert, and F. Laguillaumie, “Encryption switching protocols revisited: Switching modulo p,” in *CRYPTO 2017*, ser. LNCS, vol. 10401, 2017, pp. 255–287.
[8] G. Ateniese, K. Fu, M. Green, and S. Hohenberger, “Improved proxy re-encryption schemes with applications to secure distributed storage,” *ACM Trans. Inf. Syst. Secur.*, vol. 9, no. 1, pp. 1–30, 2006.
[9] J. Weng, Y. Zhao, and G. Hanaoka, “On the security of a bidirectional proxy re-encryption scheme from PKC 2010,” in *PKC 2011*, ser. LNCS, vol. 6571, 2011, pp. 284–295.
[10] Q. Tang, “Type-based proxy re-encryption and its construction,” in *INDOCRYPT 2008*, ser. LNCS, vol. 5365, 2008, pp. 130–144.
[11] C. Chu and W. Tzeng, “Identity-based proxy re-encryption without random oracles,” in *ISC 2007*, ser. LNCS, vol. 4779, 2007, pp. 189–202.
[12] M. Green and G. Ateniese, “Identity-based proxy re-encryption,” in *ACNS 2007*, ser. LNCS, vol. 4521, 2007, pp. 288–306.
[13] X. Liang, Z. Cao, H. Lin, and J. Shao, “Attribute based proxy re-encryption with delegating capabilities,” in *ASIACCS 2009*, 2009, pp. 276–286.
[14] T. Mizuno and H. Doi, “Hybrid proxy re-encryption scheme for attribute-based encryption,” in *Inscrypt 2009*, ser. LNCS, vol. 6151, 2009, pp. 288–302.
[15] K. Liang, M. H. Au, W. Susilo, D. S. Wong, G. Yang, and Y. Yu, “An adaptively cca-secure ciphertext-policy attribute-based proxy re-encryption for cloud data sharing,” in *ISPEC 2014*, ser. LNCS, vol. 8434, 2014, pp. 448–461.
[16] K. Liang, M. H. Au, J. K. Liu, W. Susilo, D. S. Wong, G. Yang, T. V. X. Phuong, and Q. Xie, “A dfa-based functional proxy re-encryption scheme for secure public cloud data sharing,” *IEEE Trans. Information Forensics and Security*, vol. 9, no. 10, pp. 1667–1680, 2014.
[17] J. Ning, X. Dong, Z. Cao, L. Wei, and X. Lin, “White-box traceable ciphertext-policy attribute-based encryption supporting flexible attributes,” *IEEE Trans. on Information Forensics and Security*, vol. 10, no. 6, pp. 1274–1288, 2015.
[18] J. Li, W. Yao, Y. Zhang, H. Qian, and J. Han, “Flexible and fine-grained attribute-based data storage in cloud computing,” *IEEE Trans. on Services Computing*, vol. 10, no. 5, pp. 785–796, 2017.
[19] J. Ning, Z. Cao, X. Dong, K. Liang, H. Ma, and L. Wei, “Auditable σ-time outsourced attribute-based encryption for access control in cloud computing,” *IEEE Trans. on Information Forensics and Security*, vol. 13, no. 1, pp. 94–105, 2018.
[20] J. Bethencourt, D. X. Song, and B. Waters, “New constructions and practical applications for private stream searching (extended abstract),” in *(S&P 2006)*, 2006, pp. 132–139.
[21] D. Boneh and B. Waters, “Conjunctive, subset, and range queries on encrypted data,” in *TCC 2007*, ser. LNCS, vol. 4392, 2007, pp. 535–554.
[22] S. Sedghi, P. van Liesdonk, S. Nikova, P. H. Hartel, and W. Jonker, “Searching keywords with wildcards on encrypted data,” in *SCN 2010*, ser. LNCS, vol. 6280, 2010, pp. 138–153.
[23] M. Abdalla, M. Bellare, D. Catalano, E. Kiltz, T. Kohno, T. Lange, J. Malone-Lee, G. Neven, P. Paillier, and H. Shi, “Searchable en-

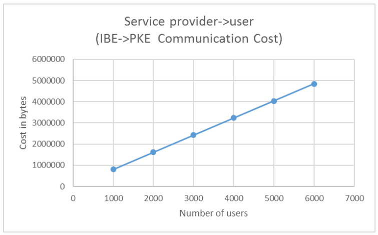


Fig. 5: Service provider →User

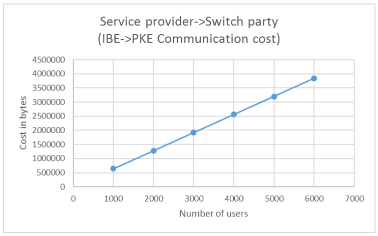


Fig. 6: Service provider→Switch party

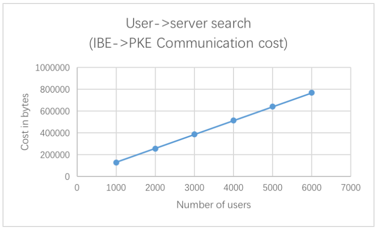


Fig. 7: User→Server-search

cryption revisited: Consistency properties, relation to anonymous IBE, and extensions," in *CRYPTO 2005*, ser. LNCS, vol. 3621, 2005, pp. 205–222.

[24] H. S. Rhee, J. H. Park, W. Susilo, and D. H. Lee, "Improved searchable public key encryption with designated tester," in *ASIACCS 2009*, 2009, pp. 376–379.

[25] P. Jiang, Y. Mu, F. Guo, and Q. Wen, "Secure-channel free keyword search with authorization in manager-centric databases," *Computers & Security*, vol. 69, pp. 50–64, 2017.

[26] P. Jiang, Y. Mu, F. Guo, X. Wang, and Q. Wen, "Online/offline ciphertext retrieval on resource constrained devices," *Comput. J.*, vol. 59, no. 7, pp. 955–969, 2016.

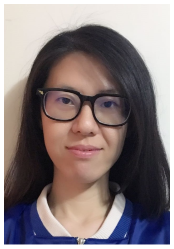
[27] J. Camenisch, M. Kohlweiss, A. Rial, and C. Sheedy, "Blind and anonymous identity-based encryption and authorised private searches on public key encrypted data," in *PKC 2009*, ser. LNCS, vol. 5443, 2009, pp. 196–214.

[28] P. Jiang, Y. Mu, F. Guo, and Q. Wen, "Public key encryption with authorized keyword search," in *ACISP 2016*, ser. LNCS, vol. 9723, 2016.

[29] J. Li, X. Lin, Y. Zhang, and J. Han, "Ksf-oabe: outsourced attribute-based encryption with keyword search function for cloud storage," *IEEE Trans. on Services Computing*, vol. 10, no. 5, pp. 715–725, 2017.

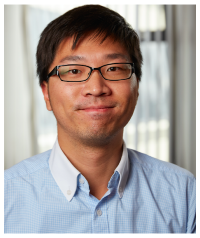
[30] J. Li, Y. Shi, and Y. Zhang, "Searchable ciphertext-policy attribute-based encryption with revocation in cloud storage," *International Journal of Communication Systems*, vol. 30, no. 1, 2017.

[31] D. Boneh, X. Boyen, and E. Goh, "Hierarchical identity based encryption with constant size ciphertext," in *EUROCRYPT 2005*, ser. LNCS, vol. 3494, 2005, pp. 440–456.



Peng Jiang received her Ph.D. degree from Beijing University of Posts and Telecommunications in 2017. She is currently an Assistant Professor in the School of Computer Science and Technology, Beijing Institute of Technology. Previously, she was a Postdoctoral Researcher in The Hong Kong Polytechnic University, Hong Kong. Her research interests include information security, applied cryptography and privacy concerns.

Jianting Ning received the Ph.D. degree from Shanghai Jiao Tong University in 2016. He is currently a research fellow at Department of Computer Science, National University of Singapore. His research interests include applied cryptography and cloud security, in particular, Public Key Encryption, Attribute-Based Encryption, and Secure Multiparty Computation.



Kaitai Liang is an Assistant Professor in Secure Systems at the University of Surrey, UK. He received the Ph.D. degree from City University of Hong Kong in 2014. His main research interests are applied cryptography, big data security, blockchain, and privacy-enhancing technology. He also has served as TPC for many renown international security/privacy conferences, e.g., ACNS, TRUSTCOM, ASIACCS, ACISP.

Changyu Dong received a Ph.D. from Imperial College London. He is currently a Senior Lecturer at the School of Computing Science, Newcastle University. His research interests include applied cryptography, trust management, data privacy and security policies. His recent work focuses mostly on designing practical secure computation protocols. The application domains include e.g. secure cloud computing and privacy preserving data mining.

Jiageng Chen received the PhD from Japan Advanced Institute of Science and Technology (JAIST) in 2012. He was working as an Assistant Professor in JAIST from 2012 to 2015. Currently, he is an Associate Professor at the School of Computer, Central China Normal University. He is the Associate Editor of Journal of Information Security and Application. His research areas include cryptography, especially algorithms, cryptanalysis, data analysis and so on.

Zhenfu Cao (SM10) received the Ph.D. degree from Harbin Institute of Technology in 1999. His research interests mainly include Number Theory, Cryptography and Information Security. He is currently a Distinguished Professor in East China Normal University. He also serves as a member of the expert panel of the National Nature Science Fund of China. He is the leader of Asia 3 Foresight Program and the key project of National Natural Science Foundation of China. He is a senior member of the IEEE.