Multi-keyword Search Employing Identity-Based Encryption Technique (MKS-IDE)

Regina Esi Turkson
School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, China.
Email: regina_turkson@yahoo.com or rturkson@ucc.edu.gh

Abstract— **Secured storage in cloud computing has been one of the challenging areas in research. Searching and retrieval of outsourced encrypted data is very difficult due to the voluminous data stored in cloud. Various strategies need to be devise to eradicate unauthorized access, retrieval and use of such data. Employing a secured scheme for an outsourced data is very relevant in order to protect the confidentiality and privacy of the data content.**

In this work, we propose a Multi-Keyword Search employing Identity-based Encryption techniques (MKS-IDE) which enables a data user to search for and retrieve encrypted files that has been outsourced into the cloud by a data owner. The retrieved encrypted files are ranked based on their relevance scores and the top-*k* relevant files are returned by the cloud to the data user. The data user then obtains a decryption key from the appropriate data owner to decrypt his selected file. The security requirement of our scheme is provably secure and the performance of our scheme is also more efficient as compared to other PEKS schemes.

Keywords— *Cloud Computing, Identity-Based Encryption (IDE), Multi-Keyword Search, Privacy Preserving*

1. INTRODUCTION

The rapid growth in cloud computing is due to it wide usage and popularity. The tremendous increase in the data stored in cloud create security and privacy issues. These challenges are not only on how to store and manage these data but also on how to effectively and efficiently analyze these data to gain insightful knowledge in making smarter decision [1][2] and also to ensure the data is secured enough.

Cloud computing is a network-based environment that centers on sharing computations or resources. The numerous benefits of cloud has accounted for different data owners outsourcing their voluminous data. IT organizations have articulated concern about critical security issues that exist with the widespread implementation of cloud computing.

Security in cloud has been one of the most argued-about issues in cloud computing making privacy preserving a very important mechanism in cloud computing. Outsourcing data into the cloud is economically attractive for the cost and complexity of long-term large-scale data storage, however, data integrity and availability is not guaranteed. Thus, to efficiently verify the correctness of the outsourced cloud data without the local copy of the data becomes a big challenge for data storage security in cloud computing. Therefore, to fully guarantee the security of the data and save the cloud user's computation resources, it is of appropriately importance to enable public auditability for cloud data storage to allow users to resort to a Third Party Auditor (TPA), who has expertise and capabilities to audit the outsourced data when needed [3].

To avoid the leakage of the outsourced data to external parties, data encryption can be adopted to alleviate this privacy concerns. Encryption ensures data confidentiality, however data search also becomes a challenge. Secure search over encrypted data was first initiated by Song et al [4] by proposing a cryptographic primitive concept called searchable encryption which enables users to perform a keyword-based search on an encrypted data, just as on plaintext data. Wang et al [5] were the first to define secure search over encrypted cloud data, however, further development made by [6][7][8][9] and more, seeks to reduce computation and storage cost and also enrich the category of search functions such as fuzzy keyword search, secure ranked multi-keyword search and similarity based search but these works are limited to single-owner model. In 1984, a public key encryption scheme was introduced by Shamir [10]. The notion of identity based cryptosystem is that the public key can be an arbitrary string. Shamir's original motivation for identity-based encryption was to simplify certificate management in e-mail systems. Since the problem was posed in 1984 there have been several proposals for IBE schemes [11][12]. Boneh and Franklin in 2001, designed the first practical identity-based cryptosystem [13]. Zhang et al [14] defined a multi-owner model for privacy preserving keyword search over encrypted cloud data, however, their scheme did not use identity based encryption. A similar system was proposed in [15] of which a cryptographic techniques, query, response randomization and ranking capability was used. However, IDE was not used in their system.

In our work, we propose a multi-keyword search in cloud that employs an identity-based encryption techniques (MKS-IDE). The framework of our scheme and its security requirements are defined. We prove that our proposed scheme satisfies the ciphertext and trapdoor indistinguishability in the random oracle [16][17]. This work adopt the ranking technique used in [14]. Finally, we demonstrate the advantage of our MKS-IDE scheme by comparing with previous PEKS schemes.

This paper is organized as follows: section II outlines the system model, threat model, and design goals. Section III states the preliminaries, Section IV outlines the algorithms and the security requirements of our scheme. We present a concrete description of our proposed MKS-IDE scheme in section V. Section VI has the security analysis. Section VII, we briefly outline the privacy preserving ranking method which we

adopted from [14], section VIII gives the performance analysis of our scheme and we conclude in section IX.

2. SYSTEM AND THREAT MODEL AND DESIGN GOALS

In this section, we describe the system model, threat model, and design goals.

2.1 System Model

In our MKS-IDE scheme, we have three entities namely; data owner, data user and cloud server. The data owner has a collection of files to be outsourced to the cloud. To ensure confidentiality, these files ought to be encrypted. For efficient and adequate search operation, the data owner builds a secure searchable index on keywords sets extracted from the files. The data owner then outsource the encrypted files together with their indexes. To facilitate keyword search over these encrypted files, a data user submits a hashed-keywords to the data owner which is then used for trapdoor creation for the data user. Upon receiving the trapdoor, the data user submits it to the cloud. The cloud server then performs secure search over the encrypted indexes from various data owners and return the corresponding set of files that matches same patterns as that in the trapdoor. The cloud then rank these files based on their relevance score and return the top-k important files to the data user. The data user can then obtain a decryption key from the data owner to decrypt the file.

2.2 Threat Model

In our threat model, we assume that the cloud server is honest but curious. The server follows our protocol but it keen to know the content of the encrypted files, keywords and the relevance scores. The data owner and data users who are authenticated by PKI are assumed to be trusted.

2.3 Design Goals

In order to ensure privacy preserving, our MKS-IDE scheme should satisfy some security and performance goals. The proposed MKS-IDE should

- Enable multi-keyword search over different encrypted files from data owners.
- Must provide data user scalability.
- Ensure that authenticated data users perform correct and appropriate search

3. PRELIMINARIES

This section gives a brief review of the various concepts of bilinear pairing, identity-based encryption and other related mathematical problems used in this paper. Due to limited page, we omit the details of these concepts.

3.1 Bilinear Pairing

3.2 Identity-Based Encryption

3.3 Other mathematically related problems and assumptions

3.3 Other mathematically related problems and assumptions.

Two mathematical hard problems are used.

- Bilinear Diffie-Hellman (BDH) problem
- Computational Diffie-Hellman (CDH) problem

4. ALGORITHMS AND SECURITY REQUIREMENTS

We define the algorithms and security requirements for our MKS-IDE by modifying the ones in [8] [10] [18]

4.1 Algorithms

An MKS-IDE scheme consist of the following polynomial time randomized seven (7) algorithms:

i. ***Setup:*** A probabilistic algorithm which takes a security parameter l as input and generates a master secret / public key pair.

ii. ***Key extract:*** This is a deterministic algorithm which takes as input a user's identity, a master secret key and system parameters and returns a user secret key sk_{ID} .

iii. ***MKS-IDE:*** This is a probabilistic algorithm which takes a public key of the data owner, a set of keywords in a document and system parameters and replies with MKS-IDE searchable ciphertext index I . The data owner encrypts each document with Symmetric-Key Encryption method using different keys for each document. The symmetric keys is then encrypted with a Public Key Encryption which has blinding capabilities.

iv. ***Trapdoor generation:*** Given multiple keywords, data user's secret key and system parameters, this probabilistic algorithm generates the trapdoor T_w .

v. ***Test:*** This deterministic algorithm takes the MKS-IDE searchable ciphertext index I , trapdoor T_w and system parameters. Upon receiving a query request T_w from the user, the cloud server match all keywords stored on it against the query request. It then extracts and return the candidate files whose indexes have same pattern as the query request.

vi. ***Ranking:*** The cloud ranks the candidate file set according to their relevance score and forward the top-*k* relevant file to the data user. An order and privacy preserving encoding scheme which encodes the relevance score to obtain the top-*k* search result will be adopted into our MKS-IDE scheme.

vii. ***Retrieval:*** This algorithm takes a blinded encrypted symmetric key and returns a blinded symmetric key for decryption of a document.

4.2 Security Requirements

Our MKS-IDE scheme must meet the following security requirements: (1) ciphertext indistinguishability and (2) trapdoor indistinquishability.

We define a security for Indistinguishability of Ciphertext from Ciphertext of Chosen Keyword Attack (IND-CC-CKA) and trapdoor indistinquishability of the MKS-IDE scheme in Game 1 and 2 involve interactions between an adversary A and a challenger C. Since there is limited pages, we omit the proof of both games.

5. A CONCRETE MKS-IDE SCHEME

The proposed scheme is a non-interactive public key encryption with keyword search which consist of seven algorithms. Details of algorithms are outlined below:

- ***Setup*:** Given l as a security parameter, a trusted Private Key Generator (PKG) can generates a master private/public key pair. Let G_1 and G_2 be an additive and multiplicative cyclic groups respectively, generated by P with a prime orders of q. The PKG generate a bilinear map $e: G_1 \times G_1 \to G_2$. The PKG picks a random number $x \in Z_q^*$ as a master secret key and computes the system's public key P_{pub} as $P_{pub} = x.P$. Let H_1, H_2 and H_3 be three cryptographic hash function defined as $H_1: \{0,1\}^* \to G_1^*$, $H_2: \{0,1\}^* \to G_1^*$ and $H_3: G_2 \to \{0,1\}^n$ where n is fixed length depending on l. The system's parameters $\{G_1, G_2, e, H_1, H_2, H_3, P, P_{pub}, n, params\}$ are published and the master secret keys x is kept secret.
- ***Key Extract*:** Given a user's identity $ID_i \in \{0,1\}^*$, the PKG returns the user's private key $sk_{ID_i} \leftarrow xQ_{ID_i}$ where $Q_{ID_i} \leftarrow H_1(ID_i)$.
- **MKS-IDE:** First computes the searchable index $I = (U, V)$ for a random $a \in Z_q^*$ where $U = aQ_{ID}$ and $V = H_3(k)$ and k is computed as $k = e(U, P_{pub}) e(H_2(w_i), P)$.
- ***Trapdoor Generation:*** Let W be the set of keywords the data user want to search for. The data user computes $t = H_2(W)$ and sends to the data owner. The data owner then returns the trapdoor $T_w = ask_{ID} + t$ to the data user.
- ***Test*:** Given I and T_w, the cloud server test for each file if $H_3(e(T_w, P) = V$ it returns the file and it id.
- ***Ranking:*** We adopt a privacy preserving ranked search scheme implement in [14] into our MKS-IDE scheme to facilitate the ranking of the candidate file to determine which file is more relevant to a certain keyword according to the encoded relevance scores.
- ***Retrieval:*** Given the IBE of the symmetric key $C = (C_1, C_2)$ where $C_1 = rP$ and $C_2 = sk.(e(P_{pub}, Q_{ID})^r)$ and $r \in Z_q^*$ used to encrypt the document. The data user blind C by computing $C' = (C)^\alpha = (C_1^\alpha, C_2^\alpha)$ and sends it to the data owner. The data owner computes $sk^\alpha = C_2^\alpha . e(r\alpha P_{pub,} Q_{ID})^{-1}$. The secret key can be deduce thereafter and be used to decrypt the encrypted document [19].

6. SECURITY ANALYSIS

We prove our MKS-IDE system is a non-interactive searchable encryption scheme that is semantically secure in a random oracle model[16][17][18] . The proof of security relies on the difficulty of the BDH and CDH problem.

6.1 Ciphertext Indistinguishability

The proposed MKS-IDE scheme satisfies ciphertext indistinguishability under an adaptive chosen plaintext attack. For the security of this scheme to be simplified, a proof of a lemma is demonstrated in which the adversary is assumed to be an outside attacker.

Lemma 6.1 *We assume an adversary A with a non-negligible probability ε_1 can break the ciphertext indistinguishability of the proposed MKS-IDE scheme in the random oracle under an adaptive chosen plaintext attack, then there exist a challenger C with a non-negligible probability $\xi \geq \frac{2\varepsilon_1}{e(q_R+1).q_{H_3}}$ who can solve the BDH problem where q_R and q_{H_3} represents the maximum numbers of making key extract and H_3 queries respectively.*

***Proof**:* The Challenger C is given an input of the BDH parameters as $\{q, G_1, G_2, e\}$ produce by G. The BDH problem is defined as: Given $(P, aP, bP, cP) = (P, P_1, P_2, P_3)$ with P being random in G_1^* and $a, b, c \in Z_q^*$ where q is the order of G_1 and G_2 as input, compute $D = e(P, P)^{abc} \in G_2$. We say that BDH is intractable if all polynomial time algorithms have a negligible advantage in solving BDH. The Challenger C find D by interacting with the adversary A as follows:

Setup: The challenger C runs the setup algorithm to generate the public parameters $\{q, G_1, G_2, e, P, P_{pub}, H_1, H_2, H_3\}$ by setting $Q_{ID} = P_2$ and $P_{pub} = P_3$. H_1, H_2 and H_3 are random oracles controlled by C. The challenger C gives the public parameters to A. The challenger C runs the Key Extract algorithm to generate A's private key sk_{ID_A} with the public key being ID_A as $sk_{ID_A} = aQ_{ID} = abP$. The challenger keeps the secret key msk and A's private key sk_{ID_A} is given to him.

H_1 queries: At any time, the adversary A can query the random oracle H_1 adaptively. To react to these queries, the Challenger C keeps a list of tuples $< ID_i, Q_i, b_i, coin_i >$, called the H_1^{LIST}. Until the adversary makes queries to the oracle, H_1^{LIST} is initially empty. The adversary A queries the oracle with ID_i, the Challenger C replies as follows:

a) If query ID_i already exist on the H_1^{LIST} in the tuple $< ID_i, Q_i, b_i, coin_i >$ then C answers with $H_1(ID_i) = Q_i \in G_1^*$

b) Otherwise C generates a random $coin_i \in \{0,1\}$ so that $\Pr[coin_i = 0] = \delta$ for some δ that will be determined later.

c) The challenger C picks a random value $b_i \in Z_q^*$.
If $coin_i = 0$, C computes $Q_i = b_i P \in G_1^*$.
If $coin_i = 1$, C computes $Q_i = b_i Q_{ID} \in G_1^*$.

d) The challenger C adds the tuple $< ID_i, Q_i, b_i, coin_i >$ to H_1^{LIST} and returns $H_1(ID_i) = Q_i$ to A. Note that Q_i is uniform in G_1^* and is independent of A's current view as required.

***H_2 queries*:** Similarly, at any time the adversary A can query the random oracle H_2. The Challenger C reacts to a query by maintaining a list of tuple $< ID_i, w_i, Qw_i, x_i >$ called the H_2^{LIST}. The list is initially empty until the adversary makes a query. When the adversary A queries the oracle H_2 for $< ID_i, w_i >$, The challenger C responds as follows:

a) If $< ID_i, w_i >$ appears in the list H_2^{LIST}, C replies with $H_2(ID_i, w_i) = Qw_i$

b) Otherwise, the challenger C selects randomly $x_i \in Z_q^*$ and computes $Qw_i = H_2(ID_i, w_i) = x_i.P$. Finally, C adds the tuple $< ID_i, w_i, Qw_i, x_i >$ in the list H_2^{LIST} and replies to the adversary A with $H_2(ID_i, w_i) = Qw_i$

H_3 queries: The adversary A can query the random oracle H_3 at any time. The challenger C responds to the query by maintaining a list of tuple $< m_i, n_i >$ called H_3^{LIST} which is initially empty. Responds to query is described as follows:
The challenger C replies to the adversary's queries to the oracle H_3 as indicated below:

a) If m_i appears in the list H_3^{LIST}, the challenger C replies with $H_3(m_i) = n_i$

b) Otherwise, the challenger C select at random $n_i \in \{0,1\}^n$ and fix $H_3(m_i) = n_i$. The challenger C then adds the pair $< m_i, n_i >$ to the H_3^{LIST} and return $H_3(m_i) = n_i$ to the adversary A.

Phase 1: Let ID_i be a private key extraction query issued by A. The Challenger C replies as follows:

a) C runs the algorithm above for responding to H_1-Queries to obtain a $Q_i \in G_1^*$ such that $H_1(ID_i) = Q_i$ and let $< ID_i, Q_i, b_i, coin_i >$ be the corresponding tuple on the H_1^{LIST}. If $coin_i = 1$, then C reports failure and abort. The attack on MKS-IDE failed.

b) If $coin_i = 0$, C gets $Q_i = b_i P$ and defines $sk_{ID_i} = b_i P_{pub} \in G_1^*$. Observe that $sk_{ID_i} = cQ_i$ and therefore sk_{ID_i} is the private key associated to public key ID_i. The Challenger then sends sk_{ID_i} to A.

Trapdoor queries: Adversary A issues a trapdoor query for keyword $< ID_i, w_i >$, the Challenger C then access the corresponding tuples $< ID_i, Q_i, b_i, coin_i >$ and $< ID_i, w_i, Qw_i, x_i >$ in the H_1^{LIST} and H_2^{LIST} respectively to computes the trapdoor $T_{w_i} = x_i sk_{ID_i} + t$ where $x_i \in Z_q^*$ and $t = H_2(w_i)$ for the adversary A.

a) *Challenge:* The adversary A sends $(ID_C^*, W_0^* W_1^*)$ to the challenger C where W_0^* and W_1^* are two challenged keywords. Upon receiving $(ID_C^*, W_0^* W_1^*)$ from A, the challenger C picks a random value $b \in \{0,1\}$ and access the corresponding tuple $< ID_c^*, W_b^*, Qw_b^*, x^* >$ in H_2^{LIST} to generate an MKS-IDE ciphertext $I^* = (U^*, V^*) = (U^*, R)$, where $R \in \{0,1\}^n$ is a random value. The restrictions remains that the adversary A did not queried a private key extraction for IDc^* and also for trapdoor query for W_0^* and W_1^*. The challenger C eventually sends I^* to the adversary A.

b) *Phase 2:* A continues with the request for key extract queries adaptively for any identity ID_C and the trapdoor query for any keyword W from the challenger C subject to the restriction that $IDc \neq IDc^*$ and $w \neq \{W_0^* \text{ or } W_1^*\}$.

c) *Guess:* Finally, the adversary A outputs a guess of a value $b' \in \{0,1\}$. A wins the game if $b' = b$.

By the assumption, the adversary A with a non-negligible probability ε_1 can distinguish the MKS-IDE ciphertext I^* under an adaptive chosen plaintext attack. Now, the challenger C picks a tuple $< m^*, n^* >$ in the H_3^{LIST} and outputs $v^* = m^* / e(H_2(w_b), P)$ as the solution for the BDH instance (P, aP, bP, cP) for $a, b, c \in Z_q^*$. It can be deduced that the output $m^* / e(H_2(w_b), P)$ is equal to $e(P,P)^{abc}$. Observe that $U = aQ_{ID}$ and $m^* = (e(U, P_{pub}) e(H_2(w_b), P))$. The challenger C accesses the corresponding tuple $< ID_c^*, W_b^*, Qw_b^*, x^* >$ in the list H_3^{LIST} and compute $(e(aQ_{ID}, cP) = e(abP, cP) = e(P,P)^{abc}$

By adopting the similar technique used in [18][20], we compute the probability of our MKS-IDE scheme by discussing the probability that the challenger C does not abort during the challenge phase. Suppose the adversary A makes q_R queries to the key extract query. In such an instance, the probability that the challenger C does not abort in phase 1 or 2 is $(\delta)^{q_R}$ and the probability that the challenger C does not terminate during the challenged step is $(1-\delta)$. Therefore, the probability that the challenger C does not terminate during the simulation is $(\delta)^{q_R} \cdot (1-\delta)^{q_R}$. This value can be maximize at

$\delta_{abt} = 1 - \frac{1}{q_R + 1}$. By using δ_{abt}, the probability that the challenger C does not abort is at least $\frac{1}{e(q_R+1)}$. Observe that the probability analysis uses the same techniques as Coron's analysis of the Full Domain Hash in [21]. The Challenger C outputs the correct D with the probability at least $\frac{2\varepsilon_1}{qH_3}$ [21] where q_{H3} represents the total number of requesting H_3 queries. Therefore, the Challenger C with a probability $\xi \geq \frac{2\varepsilon_1}{e(q_R+1).q_{H3}}$ can solve the BDH problem. This contradict to the BDH assumption.

Theorem 6.1: *The proposed non-interactive searchable encryption scheme (MKS-IDE) above is semantically secure against an adaptive chosen plaintext attack in the random oracle by satisfying the ciphertext indistinquishability under the BDH assumption.*

6.1 Trapdoor Indistinguishability

Lemma 6.2 *In the random oracle model, we assume that if there is an adversary A with a non-negligible advantage that can break the trapdoor indistinguishability of the proposed MKS-IDE scheme under the adaptive chosen keyword attack, then there exist a challenger C with a non-negligible advantage who can solve the computational CBH problem.*

Proof: The challenger C receives a CDH instance of (P, aP, bP) for $a, b \in Z_q^*$ where q is the order of G_1 and G_2. By interacting with the adversary A, the challenger C returns the CDH solution abP in Game 2. Since there is limited pages, we omit the proof for trapdoor indistinguishability since it's similar as defined in lemma 6.1.

It is assumed that the adversary A with a non-negligible advantage can differentiate the trapdoor T_{wb}^* under the adaptive chosen keyword attack. Meaning the trapdoor T_{wb}^* satisfy the equation $e(T_{wb}^*, P) = V'$ where $V' = e(U, P_{pub}).e(H_2(w_b), P)$ and Let $U = xQ_{ID}$. We can then obtain:

$$e(T_{wb}^* - H_2(w_b), P)^{x^{-1}} = e(abP, P)$$

Which implies that $e(abP, P) = e(T_{wb}^* - H_2(w_b), P)^{x^{-1}}$. Hence the Challenger can obtain $abP = x^{-1}(T_{wb}^* - H_2(w_b))$, which contradicts the CDH assumption.

Theorem 6.2 *The proposed MKS-IDE scheme is semantically secure and satisfies the trapdoor indistinquishability against an adaptive chosen plaintext attack under the CDH assumption in the random oracle.*

7. PRIVACY PRESERVING RANKED SEARCH

Our scheme adopts the privacy preserving ranked search in [14] . It is important to note that, after the retrieval of all the candidate files, the cloud server cannot return all the undifferential files to the data user due to (1) acquisition of excessive communication cost and overhead for the system if the cloud server decides to return all the candidate files. (2) the data users may only be concerned with the top-k relevant files that correspond to their queries. The scheme in [14] illustration an additive order preserving and privacy preserving encoding scheme. It then uses the encoded relevance score to obtain the top-k search result.

8. PERFORMANCE ANALYSIS AND COMPARISON

Adopting similar strategy used in [18] we analyze the performance of our MKS-IDE scheme. We compare our MKS-IDE scheme with some previously proposed PEKS schemes by concentrating more on some time consuming operations defined as follows:

- TG_e: The execution time of a bilinear map operation $e: G_1 \times G_1 \rightarrow G_2$
- TG_{mul}: The execution time for scalar multiplication operation in G_1
- TG_H: The execution time for map-to-point hash function, thus $H_1, H_2, H_3: \{0,1\}^* \rightarrow G_1$
- T_{inv}: The execution time of a modular inverse operation in Z_q

The most time consuming operation is the execution time for

TABLE 1: *Comparison between our IDRMKS and previously proposed dPEKS schemes*

	Scheme of Hwang and Lee [24]	**Scheme of Rhee et al.** [25]	**Scheme of Hu and Liu** [23]	**Scheme of Wu et al.** [18]	**Our MKS-IDE**
Public Key Setting	Pairing-based	Pairing-based	Pairing-based	ID-based	ID-based
Certificate Management	Required	Required	Required	Not Required	Not Required
Computational cost for ciphertext generation (conjunctive *n* keywords)	$(2n+2)TG_{mul}$ $+2nTG_H$	$(2n+2)TG_{mul}$ $+2nTG_H$	$(2n+2)TG_{mul}$ $+2nTG_H$	$TG_e + (n+2)TG_{mul}$ $+(n+2)TG_H$	$2TG_e + TG_{mul}$ $+(n+1)TG_H$
Computational cost for ciphertext generation (1 keywords)	$3TG_{mul}$ $+2TG_H$	$TG_e + 2TG_{mul}$ $+TG_H$	$TG_e + 2TG_{mul}$ $+TG_H$	$TG_e + 3TG_{mul}$ $+3TG_H$	$2TG_e + TG_{mul}$ $+2TG_H$
Computational cost for trapdoor generation	$3TG_{mul} + 2TG_H$ $+T_{inv}$	$3TG_{mul} + 2TG_H$ $+T_{inv}$	$3TG_{mul} + 2TG_H$ $+T_{inv}$	$2TG_{mul} + TG_H$	$TG_{mul} + TG_H$
Computational cost for test	$3TG_e$	$TG_e + 2TG_{mul}$ $+TG_H$	$TG_e + 2TG_{mul}$ $+TG_H$	$2TG_e + TG_{mul}$ $+T_{inv}$	$TG_e + 2TG_{mul}$ $+T_{inv}$

bilinear map operation TG_e as compared to the other operations stated above. In [22], the performance simulation results show that $TG_e \approx 2.5TG_{mul}$. We therefore analyze the performance of our MKS-IDE scheme for each phase. In the MKS-IDE ciphertext generation phase, it required $2TG_e + TG_{mul} + (n+1)TG_H$ to generate an MKS-IDE ciphertext, note that n represents the total number of keywords. $TG_{mul} + TG_H$ is essential to generate the trapdoor T_w in the trapdoor generation phase,

Table 1 above outlines the comparison between our MKS-IDE scheme to other dPEKS schemes [23][24][25]] in terms of public key setting and performance. From the table in [18], it can be observed that conjunctive keywords is not supported by the schemes in [23][25] and they require $nTG_e + (n+1)TG_{mul} + nTG_H$ for their ciphertext generation. We can therefore conclude from table 1 above that our MKS-IDE scheme is more efficient than the schemes in [23][24][25] with respect to ciphertext generation especially when n is sufficiently large and also efficient in the trapdoor generation. Since our MKS-IDE scheme is based on ID system, the load of certificate management associated with the other schemes which are based on pairing-based public key system are eradicated. Although the scheme in [25] is also based on ID-system, however in the ciphertext and trapdoor generation, our MKS-IDE is more efficient than [25].

9. CONCLUSION

In this paper we proposed an MKS-IDE scheme which supports conjunctive keywords. We defined the framework and the security requirements for our MKS-IDE scheme and when we compared our scheme to previous dPEKS scheme, the performance of our scheme is more efficient in both the ciphertext and trapdoor generation phase. Our ID-based system also has an advantage of eliminating certificate management associated with PKI. We also demonstrated that our MKS-IDE scheme possesses the ciphertext indistinquishability and trapdoor indistinguishability under BDH and CDH assumptions respectively.

Our future work will seek to implement our MKS-IDE scheme in a hybrid cloud and also to investigate our MKS-IDE scheme without random oracle with a pairing free algorithm to determine how feasible it will be in cloud.

REFERENCES

[1] I. Abaker, T. Hashem, I. Yaqoob, N. Badrul, S. Mokhtar, A. Gani, and S. Ullah, "The rise of ' big data ' on cloud computing : Review and open research issues," *Inf. Syst.*, vol. 47, pp. 98–115, 2015.

[2] L. Ballard, S. Kamara, and F. Monrose, "Achieving efficient conjunctive keyword searches over encrypted data," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2005, vol. 3783 LNCS, pp. 414–426.

[3] C. Wang, Q. Wang, K. Ren, and W. Lou, "Privacy-preserving public auditing for data storage security in cloud computing," *Proc. - IEEE INFOCOM*, 2010.

[4] D. Wagner, A. Perrig, D. X. Song, D. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," in *Proceeding 2000 IEEE Symposium on Security and Privacy*, 2000, pp. 44–55.

[5] P. Golle, J. Staddon, and B. Waters, "Secure conjunctive keyword search over encrypted data," *Appl. Cryptogr. Netw. Secur.*, vol. 3089, pp. 31–45, 2004.

[6] R. Li, Z. Xu, W. Kang, K. C. Yow, and C. Z. Xu, "Efficient multi-keyword ranked query over encrypted data in cloud computing," *Futur. Gener. Comput. Syst.*, vol. 30, no. 1, pp. 179–190, 2014.

[7] J. Li, Q. Wang, C. Wang, N. Cao, K. Ren, and W. Lou, "Fuzzy Keyword Search over Encrypted Data in Cloud Computing," *2010 Proc. IEEE INFOCOM*, pp. 1–5, 2010.

[8] B. Wang, S. Yu, W. Lou, and Y. T. Hou, "Privacy-preserving multi-keyword fuzzy search over encrypted data in the cloud," in *Proceedings - IEEE INFOCOM*, 2014, pp. 2112–2120.

[9] M. Chuah and W. Hu, "Privacy-aware bedtree based solution for fuzzy multi-keyword search over encrypted data," in *Proceedings - International Conference on Distributed Computing Systems*, 2011, pp. 273–281.

[10] A. Shamir, "Identity-Based Cryptosystems and Signature Schemes," *Adv. Cryptol.*, vol. 196, pp. 47–53, 1985.

[11] U. Maurer and Y. Yacobi, "Non-interactive public-key cryptography," *Adv. Cryptology—EUROCRYPT'91*, pp. 498–507, 1991.

[12] L. Problem, "An ID-Based Cryptosystem Based on the Discrete," vol. 7, no. 4, 1989.

[13] D. Boneh and M. Franklin, "Identity-Based Encryption from the Weil Pairing," *SIAM J. Comput.*, vol. 32, no. 3, pp. 586–615, 2003.

[14] W. Zhang, Y. Lin, S. Xiao, J. Wu, and S. Zhou, "Privacy Preserving Ranked Multi-Keyword Search for Multiple Data Owners in Cloud Computing," *IEEE Trans. Comput.*, vol. 65, no. 5, pp. 1566–1577, 2016.

[15] P. D. Mgmt, C. A. Secondary, and C. Author, "Distributed and Parallel Databases An Efficient Privacy-Preserving Multi-Keyword Search over Encrypted Cloud Data with."

[16] R. Canetti, O. Goldreich, and S. Halevi, "The random oracle methodology, revisited," *J. ACM*, vol. 51, no. 4, pp. 557–594, 2004.

[17] M. Bellare, "R a n d o m Oracles are Practical : A Paradigm for Designing Efficient P r o t o c o l s," pp. 62–73, 1993.

[18] T.-Y. Wu, T.-T. Tsai, and Y.-M. Tseng, "Efficient searchable ID-based encryption with a designated server," *Ann. Telecommun. - Ann. Des Télécommunications*, vol. 69, no. 7–8, pp. 391–402, 2014.

[19] C. Orencik and E. Savas, "Efficient and secure ranked multi-keyword search on encrypted cloud data," *Proc. 2012 Jt. EDBT/ICDT Work.*, pp. 186–195, 2012.

[20] Y. M. Tseng and T. T. Tsai, "Efficient revocable ID-based encryption with a public channel," *Comput. J.*, vol. 55, no. 4, pp. 475–486, 2012.

[21] J.-S. Coron, "On the Exact Security of Full Domain Hash," *CRYPTO 2000 Adv. Cryptol.*, pp. 229–235, 2000.

[22] T.-Y. Wu and Y.-M. Tseng, "An ID-Based Mutual Authentication and Key Exchange Protocol for Low-Power Mobile Devices," *Comput. J.*, vol. 53, no. 7, pp. 1062–1070, 2009.

[23] C. Hu and P. Liu, "An enhanced searchable public key encryption scheme with a designated tester and its extensions," *J. Comput.*, vol. 7, no. 3, pp. 716–723, 2012.

[24] Y. H. Hwang and P. J. Lee, "Public key encryption with conjunctive keyword search and its extension to a multi-user system," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 4575 LNCS, pp. 2–22, 2007.

[25] 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," *J. Syst. Softw.*, vol. 83, no. 5, pp. 763–771, 2010.