5-2013

Expressive Search on Encrypted Data

Junzuo LAI

Xuhua ZHOU

Robert H. DENG
Singapore Management University, robertdeng@smu.edu.sg

Yingjiu LI
Singapore Management University, yjli@smu.edu.sg

Kefei CHEN

DOI: https://doi.org/10.1145/2484313.2484345

Expressive Search on Encrypted Data

Junzuo Lai
School of Information Systems
Singapore Management University
Singapore 178902
Dept. of Computer Science
Jinan University
Guangzhou 510632, China
junzuolai@smu.edu.sg

Xuhua Zhou*
Dept. of Computer Science
Shanghai Jiao Tong University
Shanghai, China 200240
xuhuazhou2010@gmail.com

Robert H. Deng
School of Information Systems
Singapore Management University
Singapore 178902
robertdeng@smu.edu.sg

Yingjiu Li
School of Information Systems
Singapore Management University
Singapore 178902
yjli@smu.edu.sg

Kefei Chen
School of Science
Hangzhou Normal University
Zhejiang, China 310036
kfchen2004@gmail.com

ABSTRACT

Different from the traditional public key encryption, searchable public key encryption allows a data owner to encrypt his data under a user's public key in such a way that the user can generate search token keys using her secret key and then query an encryption storage server. On receiving such a search token key, the server filters all or related stored encryptions and returns matched ones as response.

Searchable pubic key encryption has many promising applications. Unfortunately, existing schemes either only support simple query predicates, such as equality queries and conjunctive queries, or have a superpolynomial blowup in ciphertext size and search token key size.

In this paper, based on the key-policy attribute-based encryption scheme proposed by Lewko *et al.* recently, we present a new construction of searchable public key encryption. Compared to previous works in this field, our construction is much more expressive and efficient and is proven secure in the standard model.

Categories and Subject Descriptors

E.3 [**Data Encryption**]: Public Key Cryptosystems; H.3 [**Information Storage and Retrieval**]: Information Search and Retrieval

*Corresponding author

ASIA CCS'13, May 8–10, 2013, Hangzhou, China.

General Terms

Design, Security

Keywords

Searchable Public Key Encryption, Public Key Encryption with Keyword Search, Anonymous Key-Policy Attribute-Based Encryption

1. INTRODUCTION

Consider a user Bob who sends email to another user Alice encrypted under Alice's public key. An email gateway is configured to check whether emails contain the keyword "urgent" so that it could route emails accordingly. On the other hand, Alice does not wish to give the gateway the ability to decrypt all her messages. Boneh *et al.* in [7] define and construct a mechanism – public key encryption with keyword search (PEKS) that allows Alice to provide a token key to the gateway so that the latter can test whether the word "urgent" is a keyword in the email while without learning anything else about the email.

In a PEKS scheme [7], to send an encrypted email message msg with keywords $s_1, \ldots, s_n$ to Alice, Bob computes

$$[E_{A_{pub}}(msg)||\mathsf{PEKS}(A_{pub}, s_1)||\cdots||\mathsf{PEKS}(A_{pub}, s_n)],$$

where A_{pub} is Alice's public key. After Alice submitting a token key $\mathrm{TK}_{s'}$ for keyword s', the email gateway can decide whether this encrypted message matches with $\mathrm{TK}_{s'}$ by testing each keyword encryption $\mathrm{PEKS}(A_{pub}, s_i)$ against $\mathrm{TK}_{s'}$ iteratively. If there exists s_i satisfying $s_i = s'$, the gateway deems that the entire encrypted message is matched and then takes appropriate actions on the email accordingly. The first part of the ciphertext above, i.e., $E_{A_{pub}}(msg)$, is normally ignored when studying a PEKS scheme, since its security can be achieved via the traditional public key encryption. Except stated explicitly, ciphertexts of a public key encryption with keyword search scheme consist of only the keyword encryptions. Therefore, informally, the basic

security requirement for a public key encryption with keyword search is that the email gateway or the storage server does not learn any information about encrypted keywords unless it has the knowledge of a matched token key.

The PEKS scheme in [7] only supports equality queries. However, more expressive search filters, e. g., conjunctive keywords or even boolean formulas, are required in many applications. For comparability, we use the email gateway application scenario in [7] to illustrate this. Two typical expressive search filters at the gateway may look like:

Eg. 1: *sender: Bob* AND *priority: urgent*

Eg. 2: (*sender: Bob* AND *priority: urgent*) OR *subject: recruitment*

where the first example means that Alice wants the email gateway to return all urgent emails sent by Bob, and the second one means that all emails either sent by Bob with urgent priority or having a subject of recruitment should be forwarded to Alice.

With regard to the conjunctive keyword search problem presented by Eg. 1, security concerns and high storage overhead (i.e., storage overhead is exponential in the number of keyword fields) overrule two straightforward solutions, namely set intersection and meta keywords [26, 16]. The set intersection is built upon simple PEKS scheme, such as the one in [7]. Given a conjunction of keywords, the gateway is provided with a search capability for every individual keyword in the conjunction. For every keyword, the gateway finds the set of ciphertexts that match that keyword, then returns the intersection of all the sets. This approach allows the gateway to learn a lot of extra information in addition to the results of the conjunctive query. The meta keyword approach is to define a meta keyword for every possible conjunction of keywords. These meta keywords are then associated with messages like regular keywords. The obvious drawback of this approach is that m keywords require 2^m meta keywords in order to accommodate all possible conjunctive queries. We refer interested readers to [26, 16] for more details.

The schemes proposed in [26, 19, 10, 20, 33] solve the conjunctive keyword search problem in the public-key setting. Especially, Boneh and Waters [10] present a general framework for analyzing and constructing searchable public key encryption (S-PKE)[1] schemes for various families of predicates. Boneh and Waters [10] then construct public key encryption schemes that support comparison queries (such as greater-than) and general subset queries. They also support arbitrary conjunctions. However, to the best of our knowledge, except S-PKE schemes constructed from inner-product predicate encryption [20], existing schemes mostly focus on conjunctive keyword search and do not work in the situations involving keyword disjunction, such as Eg. 2.

The notion of inner-product predicate encryption (IPE) is first introduced by Katz *et al.* [20]. In a predicate encryption scheme, secret keys correspond to predicates and ciphertexts are associated with a set of attributes; a secret key SK_f corresponding to a predicate f can be used to decrypt a ciphertext associated with an attribute set I if and only if $f(I) = 1$. The special case of inner product predicates is obtained by having each attributes set correspond to a vector $\vec{x}$ and each predicate $f_{\vec{v}}$ correspond to a vector $\vec{v}$, where $f_{\vec{v}}(\vec{x}) = 1$ iff $\vec{x} \cdot \vec{v} = 0$ ($\vec{x} \cdot \vec{v}$ denotes the standard inner-product). IPE can be extended to construct a solution to disjunctive keyword search. However, as shown in [20], the solution has a *superpolynomial* blowup in ciphertext size and search token key size, thus is not efficient. As described in [20], the conjunction predicate AND_{s_1,s_2} where $\mathsf{AND}_{s_1,s_2}(x_1, x_2) = 1$ iff both $x_1 = s_1$ and $x_2 = s_2$, and the disjunction predicate OR_{s_1,s_2} where $\mathsf{OR}_{s_1,s_2} = 1$ iff either $x_1 = s_1$ or $x_2 = s_2$, can be encoded as the following bi-variate polynomials

AND_{s_1,s_2}: $p'(x_1, x_2) = r \cdot (x_1 - s_1) + (x_2 - s_2)$,

OR_{s_1,s_2}: $p''(x_1, x_2) = (x_1 - s_1) \cdot (x_2 - s_2)$,

where r in the first polynomial is a random element chosen from a proper domain. Following this principle, we could easily encode Eg. 2 as a tri-variate polynomial

$$\begin{aligned}
&\text{Eg.}2_{s_1,s_2,s_3}(x_1, x_2, x_3)\\
&= (r \cdot (x_1 - s_1) + (x_2 - s_2)) \cdot (x_3 - s_3)\\
&= 0 \cdot x_1x_2x_3 + 0 \cdot x_1x_2 + r \cdot x_1x_3 + 1 \cdot x_2x_3\\
&\quad +(-rs_3) \cdot x_1 + (-s_3) \cdot x_2 + (-rs_1 - s_2) \cdot x_3\\
&\quad +s_3(rs_1 + s_2),
\end{aligned}$$

where s_1, s_2, s_3 denote "*sender: Bob*", "*priority: urgent*" and "*subject: recruitment*", respectively. To support the search type in Eg. 2, the sender Bob could take advantage of IPE to encrypt his email with the vector $\vec{x} = (x_1x_2x_3, x_1x_2, x_1x_3, x_2x_3, x_1, x_2, x_3, 1)$, where x_1, x_2, x_3 are keywords in the email; the receiver Alice generates her token key using the predicate vector $\vec{v} = (0, 0, r, 1, -rs_3, -s_3, -rs_1 - s_2, s_3(rs_1 + s_2))$ (i.e., the coefficients of the above equation). Obviously, the gateway can successfully match Alice's query to the encrypted email as long as the search filter Eg. 2 is satisfied. As pointed out in [20], the complexity of the resulting scheme is proportional to d^t, where t is the number of variables and d is the maximum degree (of the resulting polynomial) in each variable.

These facts motivate us to construct an efficient public key encryption supporting expressive search.

1.1 Our Contribution

In this paper, based on the key-policy attribute-based encryption (KP-ABE) scheme proposed by Lewko *et al.* [22] recently, we present an efficient construction of a S-PKE scheme which supports arbitrary monotone boolean predicate, such as Eg. 1 and Eg. 2. We prove that our scheme is secure in the standard model. Inheriting from [22], our scheme has the restriction that each keyword field can only be used once in a predicate.

In a KP-ABE scheme [30, 17], every ciphertext is associated with a set of attributes (i.e., keywords in S-PKE), and every user's secret key is associated with an access structure on attributes (i.e. search predicate in S-PKE). A user is able to decrypt a ciphertext only if the set of attributes associated with the ciphertext satisfies the access structure associated with the user's private key. However, the attributes (i.e., keywords) associated with ciphertexts in existing KP-ABE scheme, including Lewko *et al.* 's KP-ABE scheme [22], is not anonymous, thus a KP-ABE scheme can not be used

[1]S-PKE is a generalization of PEKS. PEKS refers to public key encryption supporting simple encrypted keyword search predicates such as equality queries while S-PKE refers to public key encryption supporting more expressive keyword search predicates.

as S-PKE. In our S-PKE scheme, the ciphertext does not reveal any information about the keywords. In fact, our S-PKE scheme can be easily extended to obtain the first efficient anonymous KP-ABE scheme, in which given a ciphertext a probabilistic polynomial time adversary cannot learn any information about the associated attribute set.

1.2 Related Work

In this section, we briefly review literature on searchable public key encryption.

Boneh *et al.* [7] initiate the research on PEKS and give a specific construction, known as the BDOP-PEKS scheme, which only supports equality queries. Abdalla *et al.* [1] formally define the property of consistency for PEKS, and state the relation between PEKS and anonymous identity based encryption (IBE). Based on different techniques or conditions, several PEKS constructions, also supporting equality queries, are presented in [13, 21, 5]. Noticing that the encryption in BDOP-PEKS and its followings is not invertible, Fuhr and Paillier [15] introduce the ability of decryption to searchable encryption (DSE) in which the receiver is allowed to decrypt the keyword ciphertext using an additional decryption algorithm.

Park *et al.* [26] propose the notion of public key encryption with conjunctive keyword search (PECK), then Hwan and Lee [19] make improvement on the sizes of ciphertext and private key, and extend the technique to multi-user setting. Zhang and Zhang [33] study a similar problem, namely conjunctive with subset keywords search (PECSK). Bringer *et al.* [11] take advantage of Bloom Filter [6] to construct an error-tolerant searchable encryption, permitting to search on encrypted data with only an approximation of some keywords. Boneh and Waters [10] present a general framework for analyzing and constructing S-PKE for various families of predicates and construct several schemes that support arbitrary conjunctions. Katz *et al.* [20] propose the notion of inner-product predicate encryption (IPE), which can be extended to construct S-PKE with disjunctive keyword search. However, as shown in [20], the resulting solution suffers from a *superpolynomial* blowup in ciphertext size and search token key size.

In addition to designing schemes with more expressive search criteria, there are also efforts [2, 34] studying the combination of a public key encryption (PKE) scheme and a PEKS scheme. Baek *et al.* [2] call their proposed scheme "PKE/PEKS" and define its security against chosen ciphertext attack (IND-PKE/PEKS-CCA). The resulting construction is based on a variation of ElGamal encryption and BDOP-PEKS, and is proved secure in the random oracle model. Zhang and Imai [34] give a generic construction which is based on secure PEKS schemes and tag-KEM/DEM schemes, and is proved secure without random oracles.

Some literature aim at enhancing the original security definition of PEKS in [7]. Byun *et al.* [12] define off-line keyword guessing (KG) attacks and show that BDOP-PEKS is insecure against KG attacks unless there exists a secure channel between the receiver and the server (i.e., the storage server or the email gateway). PEKS schemes which are immune to KG attacks (namely PEKS with a designated server, dPEKS for short [29] and its enhancement [27], or secure channel free PEKS, SCF-PEKS for short [3, 18, 14]) have also been reported in the literature. Tang and Chen [32] provide another approach to resist KG attacks, but their construction seems more like a searchable encryption scheme in the private-key setting. The security of the search token key (trapdoor), which ensures an adversary cannot learn the search criteria from a token key generated by the receiver, is discussed in [9, 28] and it is known that such a security notion can only be achieved in the private-key setting [31].

We omit the literature on searchable encryption in the private-key setting since they are outside the scope of this paper.

1.3 Organization

The rest of this paper is organized as follows. Section 2 gives some preliminaries and formally defines security of our S-PKE. Section 3 describes the proposed construction and its security proof. Section 4 concludes the paper.

2. PRELIMINARIES

If S is a set, then $s \xleftarrow{\$} S$ denotes the operation of picking an element s uniformly at random from S. Let $\mathbb{N}$ denote the set of natural numbers. If $\lambda \in \mathbb{N}$ then 1^λ denotes the string of λ ones. Let $z \leftarrow \mathsf{A}(x, y, \ldots)$ denote the operation of running an algorithm A with inputs $(x, y, \ldots)$ and output z. A function $f(\lambda)$ is *negligible* if for every $c > 0$ there exists a λ_c such that $f(\lambda) < 1/\lambda^c$ for all $\lambda > \lambda_c$.

2.1 Access Structures

Definition 1 (Access Structure [4]). *Let $\{P_1, \ldots, P_n\}$ be a set of parties. A collection $\mathbb{A} \subseteq 2^{\{P_1,\ldots,P_n\}}$ is monotone if $\forall B, C$: if $B \in \mathbb{A}$ and $B \subseteq C$, then $C \in \mathbb{A}$. An access structure (respectively, monotone access structure) is a collection (respectively, monotone collection) $\mathbb{A}$ of non-empty subsets of $\{P_1, \ldots, P_n\}$, i.e., $\mathbb{A} \subseteq 2^{\{P_1,\ldots,P_n\}} \setminus \{\emptyset\}$. The sets in $\mathbb{A}$ are called authorized sets, and the sets not in $\mathbb{A}$ are called unauthorized sets.*

In our context, keywords play the role of parties and we restrict our attention to monotone access structures (i.e., search predicates). It is possible to (inefficiently) realize general access structures using our techniques by treating the negation of a keyword as a separate keyword.

2.2 Linear Secret Sharing Schemes

Our construction will employ linear secret-sharing schemes (LSSS). We use the definition adapted from [4]:

Definition 2 (Linear Secret-Sharing Schemes). *A secret sharing scheme Π over a set of parties $\mathcal{P}$ is called linear (over $\mathbb{Z}_p$) if*

1. *The shares for each party form a vector over $\mathbb{Z}_p$.*
2. *There exists a matrix $\mathbf{A}$ with ℓ rows and n columns called the share-generating matrix for Π. For all $i = 1, \ldots, \ell$, the i^{th} row of $\mathbf{A}$ is labeled by a party $\rho(i)$ (ρ is a function from $\{1, \ldots, \ell\}$ to $\mathcal{P}$). When we consider the column vector $v = (s, r_2, \ldots, r_n)$, where $s \in \mathbb{Z}_p$ is the secret to be shared, and $r_2, \ldots, r_n \in \mathbb{Z}_p$ are randomly chosen, then $\mathbf{A}v$ is the vector of ℓ shares of the secret s according to Π. The share $(\mathbf{A}v)_i$ belongs to party $\rho(i)$.*

It is shown in [4] that every linear secret-sharing scheme according to the above definition also enjoys the linear reconstruction property, defined as follows. Suppose that Π

is an LSSS for the access structure $\mathbb{A}$. Let $S \in \mathbb{A}$ be any authorized set, and let $I \subset \{1, \ldots, \ell\}$ be defined as $I = \{i | \rho(i) \in S\}$. Then there exist constants $\{\omega_i \in \mathbb{Z}_p\}_{i \in I}$ such that, if $\{\lambda_i\}$ are valid shares of any secret s according to Π, then $\sum_{i \in I} \omega_i \lambda_i = s$. Let A_i denotes the i^{th} row of $\mathbf{A}$, we have $\sum_{i \in I} \omega_i A_i = (1, 0, \ldots, 0)$. These constants $\{\omega_i\}$ can be found in time polynomial in the size of the share-generation matrix $\mathbf{A}$ [4]. Note that, for unauthorized sets, no such constants $\{\omega_i\}$ exist.

Boolean Formulas Access structures (i.e., search predicates) might also be described in terms of monotonic boolean formulas. Using standard techniques [4] one can convert any monotonic boolean formula into an LSSS representation. We can represent the boolean formula as an access tree. An access tree of ℓ nodes will result in an LSSS matrix of ℓ rows. We refer the reader to the appendix of [24] for a discussion on how to perform this conversion.

2.3 Searchable Public Key Encryption

During the rest of the paper, we follow the general framework and notation for S-PKE as defined in [10]. Suppose a user Bob is about to send an encrypted document to Alice with keywords $s_1, \ldots, s_n$, where n is the number of keyword fields. If documents were emails for example, we could define 4 keyword fields, such as "From", "To", "Date" and "Subject". Bob sends the following message:

$$[E_{A_{pub}}[msg],\ \mathsf{S\text{-}PKE}(A_{pub}, (s_1, \ldots, s_n))],$$

where A_{pub} is Alice's public key, msg is the email body, and $\mathsf{S\text{-}PKE}$ is an algorithm with properties discussed below. To simplify the description, we ignore $E_{A_{pub}}[M]$ that can be encrypted with any secure public key encryption.

Without loss of generality, we assume that all keyword fields are defined for every document, as employed in [16]. From here onwards, we identify a document with the vector of n keywords. We denote a document by $\mathrm{D} = (s_1, \ldots, s_n)$, where s_i is the keyword of document D in the i^{th} keyword field. A S-PKE scheme consists of the following four algorithms:

$\mathsf{Setup}(1^\lambda)$ takes as input a security parameter λ and outputs a public key PK and secret key SK.

$\mathsf{Encrypt}(\mathsf{PK}, \mathrm{D} = (s_1, \ldots, s_n))$ takes as input the public key PK and a document $\mathrm{D} = (s_1, \ldots, s_n)$ and outputs a ciphertext C_{D}.

$\mathsf{GenToken}(\mathsf{PK}, \mathsf{SK}, \mathcal{P})$ takes as input the public key PK, the secret key SK and a predicate $\mathcal{P}$ and outputs a token key $\mathsf{TK}_{\mathcal{P}}$.

$\mathsf{Test}(\mathsf{PK}, \mathsf{TK}_{\mathcal{P}}, C_{\mathrm{D}})$ takes as input the public key PK, a token key $\mathsf{TK}_{\mathcal{P}} \leftarrow \mathsf{GenToken}(\mathsf{PK}, \mathsf{SK}, \mathcal{P})$ and a ciphertext $C_{\mathrm{D}} \leftarrow \mathsf{Encrypt}(\mathsf{PK}, \mathrm{D} = (s_1, \ldots, s_n))$. It outputs "yes" if the keywords $(s_1, \ldots, s_n)$ satisfies the predicate $\mathcal{P}$ (i.e., $\mathcal{P}(\mathrm{D}) = 1$) and "no" otherwise.

We now give the security model for S-PKE in the sense of semantic-security. We need to ensure that an $\mathsf{S-PKE.Encrypt}(A_{pub}, \mathrm{D} = (s_1, \ldots, s_n))$ does not reveal any information about D unless $\mathsf{TK}_{\mathcal{P}}$ with $\mathcal{P}(\mathrm{D}) = 1$ is available. We define security against an active attacker who is able to obtain token keys $\mathsf{TK}_{\mathcal{P}}$ for any $\mathcal{P}$ of his choice. Even under such attack the attacker should not be able to distinguish an encryption of a document D_0 from an encryption of a document D_1 for which he did not obtain the token key $\mathsf{TK}_{\mathcal{P}}$ such that $\mathcal{P}(\mathrm{D}_0) = 1$ or $\mathcal{P}(\mathrm{D}_1) = 1$. Formally, we define security against an active attacker $\mathcal{A}$ using the following game between a challenger and the attacker:

Setup: The challenger runs $\mathsf{Setup}(1^\lambda)$ to obtain a public key PK and secret key SK. It gives the public key PK to the adversary $\mathcal{A}$ and keeps SK to itself.

Query phase 1: The adversary $\mathcal{A}$ adaptively queries the challenger for token keys corresponding to predicates $\mathcal{P}_1, \ldots, \mathcal{P}_q$. In response, the challenger runs $\mathsf{TK}_{\mathcal{P}_i} \leftarrow \mathsf{GenToken}(\mathsf{PK}, \mathsf{SK}, \mathcal{P}_i)$ and gives the token key $\mathsf{TK}_{\mathcal{P}_i}$ to $\mathcal{A}$, for $1 \le i \le q$.

Challenge: The adversary $\mathcal{A}$ submits two documents $\mathrm{D}_0, \mathrm{D}_1$, subject to the restriction that, D_0 and D_1 cannot satisfy any of queried predicates. The challenger selects a random bit $\beta \in \{0, 1\}$, sets $C_{\mathrm{D}_\beta} \leftarrow \mathsf{Encrypt}(\mathsf{PK}, \mathrm{D}_\beta)$ and sends C_{D_β} to the adversary as its challenge ciphertext.

Query phase 2: The adversary continues to adaptively query the challenger for token keys corresponding to predicates with the added restriction that none of these can be satisfied by D_0 or D_1.

Guess: The adversary $\mathcal{A}$ outputs its guess $\beta' \in \{0, 1\}$ for β and wins the game if $\beta = \beta'$.

The advantage of the adversary in this game is defined as $|\Pr[\beta = \beta'] - 1/2|$ where the probability is taken over the random bits used by the challenger and the adversary.

Definition 3. *A S-PKE scheme is secure if all polynomial time adversaries have at most a negligible advantage in this security game.*

2.4 Composite Order Bilinear Groups

We will construct our scheme in composite order bilinear groups whose order is the product of four distinct primes. Composite order bilinear groups were first introduced in [8].

Let $\mathcal{G}$ be an algorithm that takes as input a security parameter 1^λ and outputs a tuple $(p_1, p_2, p_3, p_4, \mathbb{G}, \mathbb{G}_T, e)$, where p_1, p_2, p_3, p_4 are distinct primes, $\mathbb{G}$ and $\mathbb{G}_T$ are cyclic groups of order $N = p_1 p_2 p_3 p_4$, and $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is a map such that

1. (Bilinear) $\forall g, h \in \mathbb{G}, a, b \in \mathbb{Z}_N, e(g^a, h^b) = e(g, h)^{ab}$;
2. (Non-degenerate) $\exists g \in \mathbb{G}$ such that $e(g, g)$ has order N in $\mathbb{G}_T$.

We further require that multiplication in $\mathbb{G}$ and $\mathbb{G}_T$, as well as the bilinear map e, are computable in time polynomial in λ. We use $\mathbb{G}_{p_1}, \mathbb{G}_{p_2}, \mathbb{G}_{p_3}, \mathbb{G}_{p_4}$ to denote the subgroups of $\mathbb{G}$ having order p_1, p_2, p_3, p_4, respectively. Observe that $\mathbb{G} = \mathbb{G}_{p_1} \times \mathbb{G}_{p_2} \times \mathbb{G}_{p_3} \times \mathbb{G}_{p_4}$. Note also that if $g_1 \in \mathbb{G}_{p_1}$ and $g_2 \in \mathbb{G}_{p_2}$ then $e(g_1, g_2) = 1$. A similar rule holds whenever e is applied to elements in distinct subgroups.

We now state the complexity assumptions we use. Utilizing the theorems proposed in [20], one can easily to prove that the assumptions hold in the generic group model.

ASSUMPTION 1. *Let $\mathcal{G}$ be as above. We define the following distribution:*

$$(p_1,p_2,p_3,p_4,\mathbb{G},\mathbb{G}_T,e)\leftarrow\mathcal{G}(1^\lambda),\ N=p_1p_2p_3p_4,$$
$$g\xleftarrow{\$}\mathbb{G}_{p_1},\ X_3\xleftarrow{\$}\mathbb{G}_{p_3},\ X_4\xleftarrow{\$}\mathbb{G}_{p_4},$$
$$D=(\mathbb{G},\mathbb{G}_T,N,e,g,X_3,X_4),$$
$$T_1\xleftarrow{\$}\mathbb{G}_{p_1}\times\mathbb{G}_{p_2},\ T_2\xleftarrow{\$}\mathbb{G}_{p_1}.$$

The advantage of an algorithm $\mathcal{A}$ in breaking Assumption 1 is defined as

$$Adv^1_{\mathcal{A}}=|\Pr[\mathcal{A}(D,T_1)=1]-\Pr[\mathcal{A}(D,T_2)=1]|.$$

DEFINITION 4. *we say $\mathcal{G}$ satisfies Assumption 1 if for any polynomial time algorithm $\mathcal{A}$, $Adv^1_{\mathcal{A}}$ is negligible.*

ASSUMPTION 2. *Let $\mathcal{G}$ be as above. We define the following distribution:*

$$(p_1,p_2,p_3,p_4,\mathbb{G},\mathbb{G}_T,e)\leftarrow\mathcal{G}(1^\lambda),\ N=p_1p_2p_3p_4,$$
$$g,X_1\xleftarrow{\$}\mathbb{G}_{p_1},\ X_2,Y_2\xleftarrow{\$}\mathbb{G}_{p_2},\ X_3,Y_3\xleftarrow{\$}\mathbb{G}_{p_3},\ X_4\xleftarrow{\$}\mathbb{G}_{p_4},$$
$$D=(\mathbb{G},\mathbb{G}_T,N,e,g,X_1X_2,Y_2Y_3,X_3,X_4),$$
$$T_1\xleftarrow{\$}\mathbb{G}_{p_1}\times\mathbb{G}_{p_2}\times\mathbb{G}_{p_3},\ T_2\xleftarrow{\$}\mathbb{G}_{p_1}\times\mathbb{G}_{p_3}.$$

The advantage of an algorithm $\mathcal{A}$ in breaking Assumption 2 is defined as

$$Adv^2_{\mathcal{A}}=|\Pr[\mathcal{A}(D,T_1)=1]-\Pr[\mathcal{A}(D,T_2)=1]|.$$

DEFINITION 5. *we say $\mathcal{G}$ satisfies Assumption 2 if for any polynomial time algorithm $\mathcal{A}$, $Adv^2_{\mathcal{A}}$ is negligible.*

ASSUMPTION 3. *Let $\mathcal{G}$ be as above. We define the following distribution:*

$$(p_1,p_2,p_3,p_4,\mathbb{G},\mathbb{G}_T,e)\leftarrow\mathcal{G}(1^\lambda),\ N=p_1p_2p_3p_4,$$
$$s\xleftarrow{\$}\mathbb{Z}_N,\ g,h\xleftarrow{\$}\mathbb{G}_{p_1},g_2,X_2\xleftarrow{\$}\mathbb{G}_{p_2},$$
$$X_3\xleftarrow{\$}\mathbb{G}_{p_3},\ X_4,Z'\xleftarrow{\$}\mathbb{G}_{p_4},\ B_{24},D_{24}\xleftarrow{\$}\mathbb{G}_{p_2}\times\mathbb{G}_{p_4},$$
$$D=(\mathbb{G},\mathbb{G}_T,N,e,g,g_2,hX_2,hZ',g^sB_{24},X_3,X_4,T),$$
$$T_1=h^sD_{24},\ T_2\xleftarrow{\$}\mathbb{G}_{p_1}\times\mathbb{G}_{p_2}\times\mathbb{G}_{p_4}.$$

The advantage of an algorithm $\mathcal{A}$ in breaking Assumption 3 is defined as

$$Adv^3_{\mathcal{A}}=|\Pr[\mathcal{A}(D,T_1)=1]-\Pr[\mathcal{A}(D,T_2)=1]|.$$

DEFINITION 6. *we say $\mathcal{G}$ satisfies Assumption 3 if for any polynomial time algorithm $\mathcal{A}$, $Adv^3_{\mathcal{A}}$ is negligible.*

3. OUR PROPOSED CONSTRUCTION

Recall that a document D is identified by a vector of n keywords $(s_1,\ldots,s_n)$, where s_i is the keyword of document D in the i^{th} keyword field. For notational purposes, let i denote the i^{th} keyword field. Our S-PKE scheme supports arbitrary monotone boolean predicate. We express an monotone boolean predicate by an LSSS $(\mathbf{A},\rho,\mathcal{T})$, where $\mathbf{A}$ is a $\ell\times m$ share-generating matrix, ρ is a map from each row of $\mathbf{A}$ to a keyword field (i.e., ρ is a function from $\{1,\ldots,\ell\}$ to $\{1,\ldots,n\}$), $\mathcal{T}$ can be parsed as $(t_{\rho(1)},\ldots,t_{\rho(\ell)})$ and $t_{\rho(i)}$ is the value (i.e., keyword) of keyword field $\rho(i)$ specified by the predicate.

Using our notations, a document $\mathrm{D}=(s_1,\ldots,s_n)$ satisfies a predicate $(\mathbf{A},\rho,\mathcal{T})$ if and only if there exist $\mathcal{I}\subseteq\{1,\ldots,\ell\}$ and constants $\{\omega_i\}_{i\in\mathcal{I}}$ such that

$$\sum_{i\in\mathcal{I}}\omega_iA_i=(1,0,\ldots,0)\text{ and }s_{\rho(i)}=t_{\rho(i)}\text{ for }\forall i\in\mathcal{I},$$

where A_i denotes the i^{th} row of $\mathbf{A}$. We also say that $\mathcal{I}\subseteq\{1,\ldots,\ell\}$ satisfies $(\mathbf{A},\rho)$ if there exist constants $\{\omega_i\}_{i\in\mathcal{I}}$ such that $\sum_{i\in\mathcal{I}}\omega_iA_i=(1,0,\ldots,0)$. We define $\mathrm{I}_{\mathbf{A},\rho}$ as the set of minimum subsets of $\{1,\ldots,\ell\}$ that satisfies $(\mathbf{A},\rho)$. By "minimum", we mean the subset cannot become smaller while still satisfying $(\mathbf{A},\rho)$.

The proposed S-PKE scheme consists of the following algorithms:

$\mathsf{Setup}(1^\lambda)$**:** The setup algorithm first runs $\mathcal{G}(1^\lambda)$ to obtain $(p_1,p_2,p_3,p_4,\mathbb{G},\mathbb{G}_T,e)$ with $\mathbb{G}=\mathbb{G}_{p_1}\times\mathbb{G}_{p_2}\times\mathbb{G}_{p_3}\times\mathbb{G}_{p_4}$, where $\mathbb{G}$ and $\mathbb{G}_T$ are cyclic groups of order $N=p_1p_2p_3p_4$. Next it chooses $g,u,h_1,\ldots,h_n\in\mathbb{G}_{p_1}$, $X_3\in\mathbb{G}_{p_3}$, $X_4,Z,Z_0,Z_1,\ldots,Z_n\in\mathbb{G}_{p_4}$ and $\alpha\in\mathbb{Z}_N$ uniformly at random. The public key is published as $\mathsf{PK}=(N,gZ,e(g,g)^\alpha,U=uZ_0,\{H_i=h_i\cdot Z_i\}_{1\le i\le n},X_4)$. The secret key is $\mathsf{SK}=(g,u,h_1,\ldots,h_n,X_3,\alpha)$.

$\mathsf{Encrypt}(\mathsf{PK},\mathrm{D}=(s_1,\ldots,s_n)\in\mathbb{Z}_N^n)$**:** This encryption algorithm chooses $s\in\mathbb{Z}_N$ and $Z_{1,0},\{Z_{1,i}\}_{1\le i\le n}\in\mathbb{G}_{p_4}$ uniformly at random. The corresponding ciphertext $C_{\mathrm{D}}=(\widetilde{C},C_0,\{C_i\}_{1\le i\le n})$ is computed as

$$\widetilde{C}=e(g,g)^{\alpha s},\ C_0=(gZ)^s\cdot Z_{1,0},\ C_i=(U^{s_i}H_i)^s\cdot Z_{1,i}.$$

$\mathsf{GenToken}(\mathsf{PK},\mathsf{SK},\mathcal{P}=(\mathbf{A},\rho,\mathcal{T}))$**:** Suppose $\mathbf{A}$ is an $\ell\times m$ matrix, ρ is a map from each row A_j of $\mathbf{A}$ to $\{1,\ldots,n\}$ and $\mathcal{T}=(t_{\rho(1)},\ldots,t_{\rho(\ell)})\in\mathbb{Z}_N^\ell$. The key generation algorithm chooses a random vectors $v\in\mathbb{Z}_N^m$ such that $\mathbf{1}\cdot v=\alpha$. (Here, $\mathbf{1}$ denotes the vector with the first entry equal to 1 and the rest equal to 0). For each row A_j of $\mathbf{A}$, it chooses a random $r_j\in\mathbb{Z}_N$ and random elements $R_{1,j},R_{2,j}\in\mathbb{G}_{p_3}$. The token key $\mathsf{TK}_{\mathcal{P}}=((\mathbf{A},\rho,\mathcal{T}),\ \{K_{1,j},K_{2,j}\}_{1\le j\le\ell})$ is computed as

$$K_{1,j}=g^{A_j\cdot v}(u^{t_{\rho(j)}}h_{\rho(j)})^{r_j}R_{1,j},\ K_{2,j}=g^{r_j}R_{2,j}.$$

$\mathsf{Test}(\mathsf{PK},\mathsf{TK}_{\mathcal{P}},C_{\mathrm{D}})$**:** Let $C_{\mathrm{D}}=(\widetilde{C},C_0,\{C_i\}_{1\le i\le n})$ and $\mathsf{TK}_{\mathcal{P}}=((\mathbf{A},\rho,\mathcal{T}),\ \{K_{1,j},K_{2,j}\}_{1\le j\le\ell})$. The test algorithm first calculates $\mathrm{I}_{\mathbf{A},\rho}$ from $(\mathbf{A},\rho)$, where $\mathrm{I}_{\mathbf{A},\rho}$ denotes the set of minimum subsets of $\{1,\ldots,\ell\}$ that satisfies $(\mathbf{A},\rho)$. It then checks if there exists an $\mathcal{I}\in\mathrm{I}_{\mathbf{A},\rho}$ that satisfies

$$\widetilde{C}=\prod_{i\in\mathcal{I}}(e(C_0,K_{1,i})/e(C_{\rho(i)},K_{2,i}))^{\omega_i},$$

where $\sum_{i\in\mathcal{I}}\omega_iA_i=(1,0,\ldots,0)$. If no element in $\mathrm{I}_{\mathbf{A},\rho}$ satisfies the above equation, it outputs "no". Otherwise, it outputs "yes".

3.1 Efficiency

The size of the public key, a token key and a ciphertext are $(n+3)|\mathbb{G}|+|\mathbb{G}_T|$, $2\ell|\mathbb{G}|$ and $(n+1)|\mathbb{G}|+|\mathbb{G}_T|$, respectively, where $|\mathbb{G}|$ and $|\mathbb{G}_T|$ are the lengths of the bit-representation of a group element in $\mathbb{G}$ and $\mathbb{G}_T$ respectively. Compared with expressive searchable public key encryption scheme constructed from IPE [20], the size of a ciphertext (resp. a key) in our scheme is *linear* with the number of

keyword fields n (resp. the size of the search predicate), not *superpolynomial*.

For a search predicate $(\mathbf{A}, \rho, \mathcal{T})$, let $\iota_1 = |\mathrm{I}_{\mathbf{A},\rho}|$, $\mathrm{I}_{\mathbf{A},\rho} = \{\mathrm{I}_1, \ldots, \mathrm{I}_{\iota_1}\}$ and $\iota_2 = |\mathrm{I}_1| + \cdots + |\mathrm{I}_{\iota_1}|$. The computational costs of an encryption, and a test under $(\mathbf{A}, \rho, \mathcal{T})$ are $(n+1)t_{\mathbb{G}\_m\_e} + t_{\mathbb{G}_T\_e}$ and $\leq 2\iota_2 t_p + \iota_1 t_{\mathbb{G}_T\_m\_e}$, respectively, where t_p, $t_{\mathbb{G}_T\_e}$, $t_{\mathbb{G}\_m\_e}$ and $t_{\mathbb{G}_T\_m\_e}$ are the computational costs of bilinear map, exponentiation in $\mathbb{G}_T$, multi exponentiation in $\mathbb{G}$ and $\mathbb{G}_T$, respectively.

3.2 Discussion

Our S-PKE scheme is based on the KP-ABE scheme [22] proposed by Lewko *et al.* . Since the attributes (i.e., keywords) in Lewko *et al.* 's KP-ABE scheme [22] is not anonymous, it is not enough to obtain a secure S-PKE. On the contrary, our S-PKE scheme can be easily extended to obtain the first efficient anonymous KP-ABE scheme. We remark that the KP-ABE scheme in [22] works in a small universe of attributes, while the keywords in our S-PKE scheme have a large universe (i.e., $\mathbb{Z}_N$).

Similar to the KP-ABE scheme in [22], our proposed S-PKE scheme has the restriction that each keyword field can only be used once in a predicate, which is called one-use S-PKE. We can obtain a secure S-PKE scheme where keyword fields are used multiple times (up to a constant number of uses fixed at setup) from a one-use scheme by applying the generic transformation given in Lewko *et al.* [22]. While the transformation does incur some cost in public key and ciphertext size, it does not increase the size of the token key. Utilizing the techniques proposed by Lewko and Waters [25] recently, it is possible to eliminate the above mentioned efficiency loss and allow unrestricted use of keyword fields while still proving security in the standard model.

3.3 Security

Note that in our construction, all components of the public key (except $e(g,g)^{\alpha}$) and the ciphertext (except $\widetilde{C} = e(g,g)^{\alpha s}$) have a $\mathbb{G}_{p_4}$ part. This formation allows us to prove that the ciphertext C_{D} does not reveal any information about D unless $\mathsf{TK}_{\mathcal{P}}$ with $\mathcal{P}(\mathrm{D}) = 1$ is available; it does not affect the test algorithm, since no component in a token key has a $\mathbb{G}_{p_4}$ part. We now state the security theorem of our S-PKE scheme.

THEOREM 1. *If Assumptions 1, 2 and 3 hold, then the proposed S-PKE scheme is secure.*

PROOF. Following the approach by Lewko and Waters [23], we define two additional structures: *semi-functional* ciphertexts and *semi-functional* keys. These will not be used in the real system, but will be used in our proof.

Semi-functional Ciphertext Let g_2 denote a generator of the subgroup $\mathbb{G}_{p_2}$. A semi-functional ciphertext is created as follows. We first use the encryption algorithm to form a normal ciphertext $C_{\mathrm{D}} = (\widetilde{C}, C_0, \{C_i\}_{1\leq i\leq n})$. Then, we choose a random exponent $c \in \mathbb{Z}_N$. We also choose random values $\eta_i \in \mathbb{Z}_N$ associated to keywords. The semi-functional ciphertext C is set to be

$$\left(\widetilde{C},\ C_0 \cdot g_2^{c},\ \{C_i \cdot g_2^{c\eta_i}\}_{1\leq i\leq n}\right).$$

It should be noted that the values η_i are chosen randomly once and then fixed - these same values will also be involved in semi-functional keys which we will define below.

Semi-functional Key A semi-functional key will take on one of two forms. To create a semi-functional key, we first use the key generation algorithm to form a normal token key $\mathsf{TK}_{\mathcal{P}} = ((\mathbf{A}, \rho, \mathcal{T}),\ \{K_{1,j}, K_{2,j}\}_{1\leq j\leq \ell})$. Then, we choose random values $\gamma_j \in \mathbb{Z}_N$ associated with row j of the $\ell \times m$ matrix $\mathbf{A}$. We also choose a random vector $w \in \mathbb{Z}_N^m$ and set $\delta_j = A_j \cdot w$. The semi-functional key of type 1 is set as

$$((\mathbf{A}, \rho, \mathcal{T}),\ \{K_{1,j} \cdot g_2^{\delta_j + \gamma_j \eta_{\rho(j)}},\ K_{2,j} \cdot g_2^{\gamma_j}\}_{1\leq j\leq \ell}).$$

A semi-functional key of type 2 is formed without the terms $g_2^{\gamma_j \eta_{\rho(j)}}$ and $g_2^{\gamma_j}$ (one could also interpret this as setting $\gamma_j = 0$):

$$((\mathbf{A}, \rho, \mathcal{T}),\ \{K_{1,j} \cdot g_2^{\delta_j},\ K_{2,j}\}_{1\leq j\leq \ell}).$$

We will prove the security of our scheme based on Assumptions 1, 2 and 3 using a hybrid argument over a sequence of games. The first game, Game_{real} is the real security game (the ciphertext and all token keys are normal). In the next game, Game_0, all of token keys will be normal, but the challenge ciphertext will be semi-functional. We let q denote the number of token key queries made by the attacker. For k from 1 to q and ϱ from 1 to n, we define

$\mathsf{Game}_{k,1}$: In this game, the challenge ciphertext is semi-functional, the first $k-1$ token keys are semi-functional of type 2, the k^{th} token key is semi-functional of type 1, and the remaining token keys are normal.

$\mathsf{Game}_{k,2}$: In this game, the challenge ciphertext is semi-functional, the first k token keys are semi-functional of type 2, and the remaining token keys are normal.

$\mathsf{Game}_{\mathsf{Final}_\varrho}$: In this game, all token keys are semi-functional of type 2, and the challenge ciphertext $C_{\mathrm{D}_\beta} = (\widetilde{C}, C_0, \{C_1, \ldots, C_n\})$ is a semi-functional encryption of D_β with $C_1, \ldots, C_\varrho$, each of which is randomly chosen from $\mathbb{G}_{p_1} \times \mathbb{G}_{p_2} \times \mathbb{G}_{p_4}$.

For notational purposes, we think of $\mathsf{Game}_{0,2}$ as another way of denoting Game_0 and $\mathsf{Game}_{\mathsf{Final}_0}$ as another way of denoting $\mathsf{Game}_{q,2}$. In the final game, $\mathsf{Game}_{\mathsf{Final}_n}$, all token keys are semi-functional, and the ciphertext $C_{\mathrm{D}_\beta} = (\widetilde{C}, C_0, \{C_1, \ldots, C_n\})$ is a semi-functional encryption with $C_1, \ldots, C_n$ randomly chosen from $\mathbb{G}_{p_1} \times \mathbb{G}_{p_2} \times \mathbb{G}_{p_4}$ (thus the ciphertext is independent of D_0 and D_1 provided by the adversary). It is clear that in the final game, no adversary can have advantage greater than 0.

We prove that these games are indistinguishable in the following four lemmas. Therefore, we conclude that the advantage of the adversary in Game_{real} (i.e., the real security game) is negligible. This completes the proof of Theorem 1. □

LEMMA 1. *Suppose that $\mathcal{G}$ satisfies Assumption 1. Then Game_{real} and Game_0 are computationally indistinguishable.*

PROOF. Suppose there exists an algorithm $\mathcal{A}$ that distinguishes Game_{real} and Game_0. Then we can build an algorithm $\mathcal{B}$ with non-negligible advantage in breaking Assumption 1. $\mathcal{B}$ is given g, X_3, X_4, T and will simulate Game_{real} or Game_0 with $\mathcal{A}$. $\mathcal{B}$ chooses $Z, Z_0, Z_1, \ldots, Z_n \in \mathbb{G}_{p_4}$ and $\alpha, a_0, a_1, \ldots, a_n \in \mathbb{Z}_N$ uniformly at random. It then sets $u = g^{a_0}, h_1 = g^{a_1}, \ldots, h_n = g^{a_n}, U = uZ_0, H_1 = h_1 Z_1, \ldots, H_n = h_n Z_n$, and sends $\mathcal{A}$ the public key:

$$\mathsf{PK} = (N, gZ, e(g,g)^{\alpha}, U, \{H_i\}_{1\leq i\leq n}, X_4).$$

It can generate normal token keys in response to $\mathcal{A}$'s token key requests by using the key generation algorithm, since it knows the $\mathsf{SK} = (g, u, h_1, \ldots, h_n, X_3, \alpha)$.

At some point, $\mathcal{A}$ sends $\mathcal{B}$ two documents $\mathrm{D}_0, \mathrm{D}_1$. $\mathcal{B}$ chooses $\beta \in \{0,1\}$ randomly and does the following:

1. $\mathcal{B}$ chooses random values $\widetilde{Z}_{1,0}, \{\widetilde{Z}_{1,i}\}_{1\leq i\leq n} \in \mathbb{G}_{p_4}$.
2. Let $\mathrm{D}_\beta = (s_{\beta,1}, \ldots, s_{\beta,n})$. $\mathcal{B}$ computes
$$\widetilde{C} = e(g^\alpha, T),\ C_0 = T \cdot \widetilde{Z}_{1,0},\ C_i = T^{a_0 s_{\beta,i} + a_i} \cdot \widetilde{Z}_{1,i}.$$
3. $\mathcal{B}$ sets the challenge ciphertext as $C_{\mathrm{D}_\beta} = (\widetilde{C}, C_0, \{C_i\}_{1\leq i\leq n})$ and sends it to $\mathcal{A}$.

If $T \xleftarrow{\$} \mathbb{G}_{p_1} \times \mathbb{G}_{p_2}$, let $T = g^s g_2^c$, then
$$\widetilde{C} = e(g,g)^{\alpha s},\ C_0 = (gZ)^s Z_{1,0} \cdot g_2^c,$$
$$C_i = (U^{s_{\beta,i}} H_i)^s Z_{1,i} \cdot g_2^{c\eta_i},$$
where $Z_{1,0} = \widetilde{Z}_{1,0} Z^{-s}$, $Z_{1,i} = \widetilde{Z}_{1,i} (Z_0^{s_{\beta,i}} Z_i)^{-s}$, $\eta_i = a_0 s_{\beta,i} + a_i$. This is a semi-functional ciphertext and $\mathcal{B}$ simulates Game_0. We note that the values of $a_0, s_{\beta,i}, a_i$ modulo p_1 are uncorrelated from their values modulo p_2, so this is properly distributed. If $T \xleftarrow{\$} \mathbb{G}_{p_1}$, it is easy to observe that this is a normal ciphertext and $\mathcal{B}$ simulates Game_{real}. Hence, $\mathcal{B}$ can use the output of $\mathcal{A}$ to distinguish between these possibilities for T. □

Lemma 2. *Suppose that $\mathcal{G}$ satisfies Assumption 2. Then $\mathsf{Game}_{k-1,2}$ and $\mathsf{Game}_{k,1}$ are computationally indistinguishable.*

Proof. Suppose there exists an algorithm $\mathcal{A}$ that distinguishes $\mathsf{Game}_{k-1,2}$ and $\mathsf{Game}_{k,1}$. Then we can build an algorithm $\mathcal{B}$ with non-negligible advantage in breaking Assumption 2. $\mathcal{B}$ is given $g, X_1X_2, Y_2Y_3, X_3, X_4, T$ and will simulate $\mathsf{Game}_{k-1,2}$ or $\mathsf{Game}_{k,1}$ with $\mathcal{A}$. $\mathcal{B}$ chooses $Z, Z_0, Z_1, \ldots, Z_n \in \mathbb{G}_{p_4}$ and $\alpha, a_0, a_1, \ldots, a_n \in \mathbb{Z}_N$ uniformly at random. It then sets $u = g^{a_0}, h_1 = g^{a_1}, \ldots, h_n = g^{a_n}, U = uZ_0, H_1 = h_1 Z_1, \ldots, H_n = h_n Z_n$, and sends $\mathcal{A}$ the public key:
$$\mathsf{PK} = (N, gZ, e(g,g)^\alpha, U, \{H_i\}_{1\leq i\leq n}, X_4).$$
Note that $\mathcal{B}$ knows the secret key $\mathsf{SK} = (g, u, h_1, \ldots, h_n, X_3, \alpha)$ associated with PK. Let us now explain how $\mathcal{B}$ answers the j^{th} key query for a predicate $(\mathbf{A}, \rho, \mathcal{T} = (t_{\rho(1)}, \ldots, t_{\rho(\ell)}))$.

For $j < k$, $\mathcal{B}$ creates a semi-functional key of type 2 by choosing a random vector v such that $\mathbf{1} \cdot v = \alpha$, a random vector w', random exponents $r_i \in \mathbb{Z}_N$, random elements $R_{1,i}, R_{2,i} \in \mathbb{G}_{p_3}$, and setting:
$$K_{1,i} = g^{A_i \cdot v} (u^{t_{\rho(i)}} h_{\rho(i)})^{r_i} R_{1,i} (Y_2Y_3)^{A_i \cdot w'},\ K_{2,i} = g^{r_i} R_{2,i}.$$
We note that this is a properly distributed semi-functional key of type 2 because the value of $A_i \cdot w'$ modulo p_2 is uncorrelated to its value modulo p_3.

For $j > k$, $\mathcal{B}$ creates a normal token key by running the key generation algorithm since it knows SK.

To answer the k^{th} key quest for $(\mathbf{A}, \rho, \mathcal{T} = (t_{\rho(1)}, \ldots, t_{\rho(\ell)}))$, $\mathcal{B}$ chooses a random vector v' such that $v' \cdot \mathbf{1} = \alpha$, a random vector w such that $w \cdot \mathbf{1} = 0$, random exponents $\tilde{\gamma}_i \in \mathbb{Z}_N$, random elements $\widetilde{R}_{1,i}, \widetilde{R}_{2,i} \in \mathbb{G}_{p_3}$ and sets:
$$K_{1,i} = g^{A_i \cdot v'} T^{A_i \cdot w} T^{\tilde{\gamma}_i (a_0 t_{\rho(i)} + a_{\rho(i)})} \widetilde{R}_{1,i},\ K_{2,i} = T^{\tilde{\gamma}_i} \widetilde{R}_{2,i}.$$
We have the following observations. If $T \xleftarrow{\$} \mathbb{G}_{p_1} \times \mathbb{G}_{p_2} \times \mathbb{G}_{p_3}$, then T can be written as $g^r g_2^d R$, and
$$K_{1,i} = g^{A_i \cdot v} (u^{t_{\rho(i)}} h_{\rho(i)})^{r_i} R_{1,i} \cdot g_2^{\delta_i + \gamma_i \eta_{\rho(i)}},$$
$$K_{2,i} = g^{r_i} R_{2,i} \cdot g_2^{\gamma_i},$$
where $v = v' + rw$, $r_i = r\tilde{\gamma}_i$, $\delta_i = dA_i w$, $\gamma_i = d\tilde{\gamma}_i$, $R_{1,i} = R^{A_i w + \tilde{\gamma}_i (a_0 t_{\rho(i)} + a_{\rho(i)})} \widetilde{R}_{1,i}$, $R_{2,i} = R^{\tilde{\gamma}_i} \widetilde{R}_{2,i}$, $\eta_{\rho(i)} = a_0 t_{\rho(i)} + a_{\rho(i)}$. This is a semi-function key of type 1. Note that the values of $\tilde{\gamma}_i, a_0, t_{\rho(i)}, a_{\rho(i)}$ modulo p_1 are uncorrelated from their values modulo p_2. If $T \xleftarrow{\$} \mathbb{G}_{p_1} \times \mathbb{G}_{p_3}$, this is a properly distributed normal token key.

At some point, $\mathcal{A}$ sends $\mathcal{B}$ two documents $\mathrm{D}_0, \mathrm{D}_1$. $\mathcal{B}$ chooses $\beta \in \{0,1\}$ randomly and does the following:

1. $\mathcal{B}$ chooses random values $\widetilde{Z}_{1,0}, \{\widetilde{Z}_{1,i}\}_{1\leq i\leq n} \in \mathbb{G}_{p_4}$.
2. Let $\mathrm{D}_\beta = (s_{\beta,1}, \ldots, s_{\beta,n})$. $\mathcal{B}$ computes
$$\widetilde{C} = e(g^\alpha, X_1X_2),\ C_0 = (X_1X_2) \cdot \widetilde{Z}_{1,0},$$
$$C_i = (X_1X_2)^{a_0 s_{\beta,i} + a_i} \cdot \widetilde{Z}_{1,i}.$$
3. $\mathcal{B}$ sets the challenge ciphertext as $C_{\mathrm{D}_\beta} = (\widetilde{C}, C_0, \{C_i\}_{1\leq i\leq n})$ and sends it to $\mathcal{A}$.

If we let $X_1X_2 = g^s g_2^c$, then
$$\widetilde{C} = e(g,g)^{\alpha s},\ C_0 = (gZ)^s Z_{1,0} \cdot g_2^c,$$
$$C_i = (U^{s_{\beta,i}} H_i)^s Z_{1,i} \cdot g_2^{c\eta_i},$$
where $Z_{1,0} = \widetilde{Z}_{1,0} Z^{-s}$, $Z_{1,i} = \widetilde{Z}_{1,i} (Z_0^{s_{\beta,i}} Z_i)^{-s}$, $\eta_i = a_0 s_{\beta,i} + a_i$. This is a semi-functional ciphertext. Note that the values of $a_0, s_{\beta,i}, a_i$ modulo p_1 are uncorrelated from their values modulo p_2.

Similar to the analysis in the proof of Lewko *et al.* 's KP-ABE scheme [22], the k^{th} key and the challenge ciphertext are properly distributed. We can thus conclude that, if $T \xleftarrow{\$} \mathbb{G}_{p_1} \times \mathbb{G}_{p_2} \times \mathbb{G}_{p_3}$, then $\mathcal{B}$ has properly simulated $\mathsf{Game}_{k,1}$. If $T \xleftarrow{\$} \mathbb{G}_{p_1} \times \mathbb{G}_{p_3}$, then $\mathcal{B}$ has properly simulated $\mathsf{Game}_{k-1,2}$. Hence, $\mathcal{B}$ can use the output of $\mathcal{A}$ to distinguish between these possibilities for T. □

Lemma 3. *Suppose that $\mathcal{G}$ satisfies Assumption 2. Then $\mathsf{Game}_{k,1}$ and $\mathsf{Game}_{k,2}$ are computationally indistinguishable.*

Proof. Suppose there exists an algorithm $\mathcal{A}$ that distinguishes $\mathsf{Game}_{k,1}$ and $\mathsf{Game}_{k,2}$. Then we can build an algorithm $\mathcal{B}$ with non-negligible advantage in breaking Assumption 2. $\mathcal{B}$ is given $g, X_1X_2, Y_2Y_3, X_3, X_4, T$ and will simulate $\mathsf{Game}_{k,1}$ or $\mathsf{Game}_{k,2}$ with $\mathcal{A}$. $\mathcal{B}$ chooses $Z, Z_0, Z_1, \ldots, Z_n \in \mathbb{G}_{p_4}$ and $\alpha, a_0, a_1, \ldots, a_n \in \mathbb{Z}_N$ uniformly at random. It then sets $u = g^{a_0}, h_1 = g^{a_1}, \ldots, h_n = g^{a_n}, U = uZ_0, H_1 = h_1 Z_1, \ldots, H_n = h_n Z_n$, and sends $\mathcal{A}$ the public key:
$$\mathsf{PK} = (N, gZ, e(g,g)^\alpha, U, \{H_i\}_{1\leq i\leq n}, X_4).$$

The responses to all key queries and challenge ciphertexts are the same as in Lemma 2, except that the k^{th} query which is given below.

To answer the k^{th} key quest for $(\mathbf{A}, \rho, \mathcal{T} = (t_{\rho(1)}, \ldots, t_{\rho(\ell)}))$, $\mathcal{B}$ chooses a random vector v such that $v \cdot \mathbf{1} = \alpha$, a random vector w, random exponents $\tilde{\gamma}_i \in \mathbb{Z}_N$, random elements $\widetilde{R}_{1,i}, \widetilde{R}_{2,i} \in \mathbb{G}_{p_3}$ and sets:
$$K_{1,i} = g^{A_i \cdot v} (Y_2Y_3)^{A_i \cdot w} T^{\tilde{\gamma}_i (a_0 t_{\rho(i)} + a_{\rho(i)})} \widetilde{R}_{1,i},\ K_{2,i} = T^{\tilde{\gamma}_i} \widetilde{R}_{2,i}.$$

We have the following observations. If $T \xleftarrow{\$} \mathbb{G}_{p_1} \times \mathbb{G}_{p_2} \times \mathbb{G}_{p_3}$, then T can be written as $g^r g_2^d R$, and

$$K_{1,i} = g^{A_i \cdot v} (u^{t_{\rho(i)}} h_{\rho(i)})^{r_i} R_{1,i} \cdot g_2^{\delta_i + \gamma_i \eta_{\rho(i)}},$$
$$K_{2,i} = g^{r_i} R_{2,i} \cdot g_2^{\gamma_i},$$

where $r_i = r\tilde{\gamma}_i$, $\delta_i = \log_{g_2} Y_2 \cdot A_i w$, $\gamma_i = d\tilde{\gamma}_i$, $R_{1,i} = Y_3^{A_i \cdot w} R^{A_i w + \tilde{\gamma}_i (a_0 t_{\rho(i)} + a_{\rho(i)})} \widetilde{R}_{1,i}$, $R_{2,i} = R^{\tilde{\gamma}_i} \widetilde{R}_{2,i}$, $\eta_{\rho(i)} = a_0 \cdot t_{\rho(i)} + a_{\rho(i)}$. This is a semi-function key of type 1. Note that the values of $\tilde{\gamma}_i, a_0, t_{\rho(i)}, a_{\rho(i)}$ modulo p_1 are uncorrelated from their values modulo p_2. If $T \xleftarrow{\$} \mathbb{G}_{p_1} \times \mathbb{G}_{p_3}$, this is a properly distributed semi-functional key of type 2.

We can conclude that, if $T \xleftarrow{\$} \mathbb{G}_{p_1} \times \mathbb{G}_{p_2} \times \mathbb{G}_{p_3}$, then $\mathcal{B}$ has properly simulated $\mathsf{Game}_{k,1}$. If $T \xleftarrow{\$} \mathbb{G}_{p_1} \times \mathbb{G}_{p_3}$, then $\mathcal{B}$ has properly simulated $\mathsf{Game}_{k,2}$. Hence, $\mathcal{B}$ can use the output of $\mathcal{A}$ to distinguish between these possibilities for T. □

LEMMA 4. *Suppose that $\mathcal{G}$ satisfies Assumption 3. Then $\mathsf{Game}_{\mathsf{Final}_{\varrho-1}}$ and $\mathsf{Game}_{\mathsf{Final}_{\varrho}}$ are computationally indistinguishable.*

PROOF. Suppose there exists an algorithm $\mathcal{A}$ that distinguishes $\mathsf{Game}_{\mathsf{Final}_{\varrho-1}}$ and $\mathsf{Game}_{\mathsf{Final}_{\varrho}}$. Then we can build an algorithm $\mathcal{B}$ with non-negligible advantage in breaking Assumption 3. $\mathcal{B}$ is given $(g, g_2, hX_2, hZ', g^s B_{24}, X_3, X_4, T)$ and will simulate $\mathsf{Game}_{\mathsf{Final}_{\varrho-1}}$ or $\mathsf{Game}_{\mathsf{Final}_{\varrho}}$ with $\mathcal{A}$. $\mathcal{B}$ chooses $Z, Z_0, Z_1, \ldots, Z_{\max\{1,\varrho-1\}}, Z_{\varrho+1} \ldots, Z_n \in \mathbb{G}_{p_4}$ and $\alpha, a_0, a_1, \ldots, a_{\max\{1,\varrho-1\}}, a_{\varrho+1} \ldots, a_n \in \mathbb{Z}_N$ uniformly at random. It then sets $u = g^{a_0}, h_1 = g^{a_1}, \ldots, h_{\max\{1,\varrho-1\}} = g^{\max\{1,\varrho-1\}}$, $h_{\varrho+1} = g^{\varrho+1} \ldots, h_n = g^{a_n}, U = uZ_0$, $H_1 = h_1 Z_1$, …, $H_{\max\{1,\varrho-1\}} = h_{\max\{1,\varrho-1\}} \cdot Z_{\max\{1,\varrho-1\}}$, $H_\varrho = hZ'$ (implicitly setting $h_\varrho = h$ and $Z_\varrho = Z'$), $H_{\varrho+1} = h_{\varrho+1} Z_{\varrho+1}, \ldots$, $H_n = h_n Z_n$, and sends $\mathcal{A}$ the public key:

$$\mathsf{PK} = (N, gZ, e(g,g)^\alpha, U, \{H_i\}_{1 \le i \le n}, X_4).$$

Each time $\mathcal{B}$ is asked to provide a key for $(\mathbf{A}, \rho, \mathcal{T} = (t_{\rho(1)}, \ldots, t_{\rho(\ell)}))$ (where $\mathbf{A}$ is an $\ell \times m$ matrix), $\mathcal{B}$ creates a semi-functional key of type 2 by choosing a random vector v such that $\mathbf{1} \cdot v = \alpha$, a random vector w, random exponents $r_i \in \mathbb{Z}_N$, random elements $R_{1,i}, R_{2,i} \in \mathbb{G}_{p_3}$, and setting:

$$K_{1,i} = \begin{cases} g^{A_i \cdot v} (g^{a_0 t_{\rho(i)}} (hX_2))^{r_i} R_{1,i} g_2^{A_i \cdot w}, & \text{if } \rho(i) = \varrho; \\ g^{A_i \cdot v} (g^{a_0 t_{\rho(i)} + a_{\rho(i)}})^{r_i} R_{1,i} g_2^{A_i \cdot w}, & \text{otherwise.} \end{cases}$$
$$K_{2,i} = g^{r_i} R_{2,i}.$$

We note that $K_{1,i}$ can be written as $g^{A_i \cdot v} (u^{t_{\rho(i)}} h_{\rho(i)})^{r_i} R_{1,i} \cdot g_2^{\delta_i}$, where $\delta_i = A_i \cdot w + r_i \log_{g_2} X_2$ if $\rho(i) = \varrho$ or $\delta_i = A_i \cdot w$, so this is a properly distributed semi-functional key of type 2.

At some point, $\mathcal{A}$ sends $\mathcal{B}$ two documents $\mathrm{D}_0, \mathrm{D}_1$. $\mathcal{B}$ chooses $\beta \in \{0,1\}$ randomly and does the following:

1. $\mathcal{B}$ chooses random values $\widetilde{Z}_{1,0}, \{\widetilde{Z}_{1,i}\}_{\varrho \le i \le n} \in \mathbb{G}_{p_4}$.
2. Let $\mathrm{D}_\beta = (s_{\beta,1}, \ldots, s_{\beta,n})$. $\mathcal{B}$ chooses random elements $C_1, \ldots, C_{\max\{1,\varrho-1\}} \in \mathbb{G}_{p_1} \times \mathbb{G}_{p_2} \times \mathbb{G}_{p_4}$ and computes
$$\widetilde{C} = e(g, g^s B_{24})^\alpha, \; C_0 = g^s B_{24} \cdot \widetilde{Z}_{1,0},$$
$$C_\varrho = (g^s B_{24})^{a_0 s_{\beta,\varrho}} \cdot T \cdot \widetilde{Z}_{1,\varrho},$$
$$\{C_i = (g^s B_{24})^{a_0 s_{\beta,i} + a_i} \cdot \widetilde{Z}_{1,i}\}_{\varrho < i \le n}.$$
3. $\mathcal{B}$ sets the challenge ciphertext as $C_{\mathrm{D}_\beta} = (\widetilde{C}, C_0, \{C_i\}_{1 \le i \le n})$ and sends it to $\mathcal{A}$.

Let B_2, B_4 be the $\mathbb{G}_{p_2}, \mathbb{G}_{p_4}$ parts of B_{24} respectively. If $T = h^s D_{24}$, then

$$\widetilde{C} = e(g,g)^{\alpha s}, \; C_0 = (gZ)^s Z_{1,0} \cdot g_2^c,$$
$$\{C_i = (U^{s_{\beta,i}} H_i)^s Z_{1,i} \cdot g_2^{c\eta_i}\}_{\varrho \le i \le n},$$

where $c = \log_{g_2} B_2$, $Z_{1,0} = B_4 \widetilde{Z}_{1,0} Z^{-s}$, $Z_{1,\varrho} = B_4^{a_0 s_{\beta,\varrho}} D_4$ $\widetilde{Z}_{1,\varrho} (Z_0^{s_{\beta,\varrho}} Z')^{-s}$, $\{Z_{1,i} = B_4^{a_0 s_{\beta,i} + a_i} \widetilde{Z}_{1,i} (Z_0^{s_{\beta,i}} Z_i)^{-s}\}_{\varrho < i \le n}$, $\eta_\varrho = (\log_{g_2} (B_2^{a_0 s_{\beta,\varrho}} D_2))/c$, $\{\eta_i = a_0 s_{\beta,i} + a_i\}_{\varrho < i \le n}$, D_2, D_4 are the $\mathbb{G}_{p_2}, \mathbb{G}_{p_4}$ parts of D_{24} respectively. Note that the values of $a_0, s_{\beta,i}, a_i$ modulo p_1 are uncorrelated to their values modulo p_2. This is a properly distributed semi-functional ciphertext with $C_1, \ldots, C_{\max\{1,\varrho-1\}}$ random in $\mathbb{G}_{p_1} \times \mathbb{G}_{p_2} \times \mathbb{G}_{p_4}$. If $T \xleftarrow{\$} \mathbb{G}_{p_1} \times \mathbb{G}_{p_2} \times \mathbb{G}_{p_4}$, this is a properly distributed semi-functional ciphertext with $C_1, \ldots, C_\varrho$ randomly chosen from $\mathbb{G}_{p_1} \times \mathbb{G}_{p_2} \times \mathbb{G}_{p_4}$.

We can conclude that, if $T = h^s D_{24}$, then $\mathcal{B}$ has properly simulated $\mathsf{Game}_{\mathsf{Final}_{\varrho-1}}$. If $T \xleftarrow{\$} \mathbb{G}_{p_1} \times \mathbb{G}_{p_2} \times \mathbb{G}_{p_4}$, then $\mathcal{B}$ has properly simulated $\mathsf{Game}_{\mathsf{Final}_{\varrho}}$. Hence, $\mathcal{B}$ can use the output of $\mathcal{A}$ to distinguish between these possibilities for T. □

4. CONCLUSIONS

Based on the KP-ABE scheme proposed by Lewko *et al.* [22] recently, we presented a new construction of searchable public key encryption. Our scheme supports arbitrary monotone boolean predicates. Compared to previous works in this field, our construction is more expressive and efficient and is proven secure in the standard model.

5. ACKNOWLEDGEMENTS

The authors thank the anonymous reviewers for their helpful comments. This work is in part supported by the Office of Research, Singapore Management University. The first author is partially supported by Natural Science Foundation of China (No. 61272534). The second and fifth authors are supported by National Science Foundation of China (No. 61133014).

6. REFERENCES

[1] M. Abdalla, M. Bellare, D. Catalano, E. Kiltz, T. Kohno, T. Lange, J. Malone-Lee, G. Neven, P. Paillier, and H. Shi. Searchable encryption revisited: Consistency properties, relation to anonymous ibe, and extensions. In *CRYPTO*, pages 205–222, 2005.

[2] J. Baek, R. Safavi-Naini, and W. Susilo. On the integration of public key data encryption and public key encryption with keyword search. In *ISC*, pages 217–232, 2006.

[3] J. Baek, R. Safavi-Naini, and W. Susilo. Public key encryption with keyword search revisited. In *ICCSA (1)*, pages 1249–1259, 2008.

[4] A. Beimel. *Secure Schemes for Secret Sharing and Key Distribution*. PhD thesis, Israel Institute of Technology, 1996.

[5] M. Bellare, A. Boldyreva, and A. O'Neill. Deterministic and efficiently searchable encryption. In *CRYPTO*, pages 535–552, 2007.

[6] B. H. Bloom. Space/time trade-offs in hash coding with allowable errors. *Commun. ACM*, 13(7):422–426, 1970.

[7] D. Boneh, G. D. Crescenzo, R. Ostrovsky, and G. Persiano. Public key encryption with keyword search. In *EUROCRYPT*, pages 506–522, 2004.

[8] D. Boneh, E.-J. Goh, and K. Nissim. Evaluating 2-dnf formulas on ciphertexts. In *TCC*, pages 325–341, 2005.

[9] D. Boneh, E. Kushilevitz, R. Ostrovsky, and W. E. S. III. Public key encryption that allows pir queries. In *CRYPTO*, pages 50–67, 2007.

[10] D. Boneh and B. Waters. Conjunctive, subset, and range queries on encrypted data. In *TCC*, pages 535–554, 2007.

[11] J. Bringer, H. Chabanne, and B. Kindarji. Error-tolerant searchable encryption. In *ICC*, pages 1–6, 2009.

[12] J. W. Byun, H. S. Rhee, H.-A. Park, and D. H. Lee. Off-line keyword guessing attacks on recent keyword search schemes over encrypted data. In *Secure Data Management*, pages 75–83, 2006.

[13] G. D. Crescenzo and V. Saraswat. Public key encryption with searchable keywords based on jacobi symbols. In *INDOCRYPT*, pages 282–296, 2007.

[14] L. Fang, W. Susilo, C. Ge, and J. Wang. A secure channel free public key encryption with keyword search scheme without random oracle. In *CANS*, pages 248–258, 2009.

[15] T. Fuhr and P. Paillier. Decryptable searchable encryption. In *ProvSec*, pages 228–236, 2007.

[16] P. Golle, J. Staddon, and B. R. Waters. Secure conjunctive keyword search over encrypted data. In *ACNS*, pages 31–45, 2004.

[17] V. Goyal, O. Pandey, A. Sahai, and B. Waters. Attribute-based encryption for fine-grained access control of encrypted data. In *ACM Conference on Computer and Communications Security*, pages 89–98, 2006.

[18] C. Gu, Y. Zhu, and H. Pan. Efficient public key encryption with keyword search schemes from pairings. In *Inscrypt*, pages 372–383, 2007.

[19] Y. H. Hwang and P. J. Lee. Public key encryption with conjunctive keyword search and its extension to a multi-user system. In *Pairing*, pages 2–22, 2007.

[20] J. Katz, A. Sahai, and B. Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. *IACR Cryptology ePrint Archive*, 2007:404, 2007.

[21] D. Khader. Public key encryption with keyword search based on k-resilient ibe. In *ICCSA (3)*, pages 298–308, 2006.

[22] A. B. Lewko, T. Okamoto, A. Sahai, K. Takashima, and B. Waters. Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption. In *EUROCRYPT*, pages 62–91, 2010.

[23] A. B. Lewko and B. Waters. New techniques for dual system encryption and fully secure hibe with short ciphertexts. In *TCC*, pages 455–479, 2010.

[24] A. B. Lewko and B. Waters. Decentralizing attribute-based encryption. In *EUROCRYPT*, pages 568–588, 2011.

[25] A. B. Lewko and B. Waters. New proof methods for attribute-based encryption: Achieving full security through selective techniques. In *CRYPTO*, pages 180–198, 2012.

[26] D. J. Park, K. Kim, and P. J. Lee. Public key encryption with conjunctive field keyword search. In *WISA*, pages 73–86, 2004.

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

[28] H. S. Rhee, J. H. Park, W. Susilo, and D. H. Lee. Trapdoor security in a searchable public-key encryption scheme with a designated tester. *Journal of Systems and Software*, 83(5):763–771, 2010.

[29] H. S. Rhee, W. Susilo, and H.-J. Kim. Secure searchable public key encryption scheme against keyword guessing attacks. *IEICE Electronics Express*, 6(5):237–243, 2009.

[30] A. Sahai and B. Waters. Fuzzy identity-based encryption. In *EUROCRYPT*, pages 457–473, 2005.

[31] E. Shen, E. Shi, and B. Waters. Predicate privacy in encryption systems. In *TCC*, pages 457–473, 2009.

[32] Q. Tang and L. Chen. Public-key encryption with registered keyword search. In *EuroPKI*, pages 163–178, 2009.

[33] B. Zhang and F. Zhang. An efficient public key encryption with conjunctive-subset keywords search. *J. Network and Computer Applications*, 34(1):262–267, 2011.

[34] R. Zhang and H. Imai. Generic combination of public key encryption with keyword search and public key encryption. In *CANS*, pages 159–174, 2007.