Indonesian Journal of Electrical Engineering and Computer Science
Vol. 4, No. 3, December 2016, pp. 684 ~ 694
DOI: 10.11591/ijeecs.v4.i3.pp684-694

Distributed Searchable Asymmetric Encryption

Shoulin Yin, Lin Teng*, Jie Liu
Software College, Shenyang Normal University, China
*Corresponding author, e-mail: 1532554069@qq.com

Abstract

Searchable asymmetric encryption (SAE) can also be called Public Key Encryption with Keyword Search (PEKS), which allows us to search the keyword on the data of having been encrypted. The essence of Asymmetric searchable encryption is that users exchange the data of encryption, one party sends a ciphertext with key encryption, and the other party with another key receives the ciphertext. Encryption key is not the same as the decryption key, and cannot deduce another key from any one of the key, thus it greatly enhances the information protection, and can prevent leakage the user's search criteria-Search Pattern. Secure schemes of SAE are practical, sometimes; however the speed of encryption/decryption in Public-key encryption is slower than private key. In order to get higher efficiency and security in information retrieval, in this paper we introduce the concept of distributed SAE, which is useful for security and can enable search operations on encrypted data. Moreover, we give the proof of security.

Keywords*: asymmetric searchable encryption, PEKS, search pattern, DSAE.*

1. Introduction

AASD In the 21st century, with the rapid development of communication technology, cloud service has entered the large number of people's live and work. Exposing the user data security of the third party service providers leads to security issues. To protect user's data privacy has become more and more important and urgent, which requires encryption. However, the cloud service that its characteristics of convenient and flexible way to charge, more and more users choose the local data migration to the cloud server. Many netters delegate to a third party provider or service provider right to search for their data. There are many scholars having done some research about the Public Key Encryption with Keyword Search [1-4].

Because under the management of the cloud server, it saves large consumption of data management and system maintenance. The data stores in the cloud without the user's physical management that cloud server administrators or illegal users (such as hackers who doesn't have access to scan) can try to get data that it's in an attempt to obtain the information contained in the data. It will cause a mass of user's data information and privacy to leak. In recent years, due to the illegal invasion of hackers and the misconduct of cloud server administrator cause some cloud security accidents, directly led to a large number of users' information and personal data leaked. For example, in 2011, hackers broke into Sony company which resulted in hundreds of millions of user data leaked [5] and Google Gmail large-scale user data were leaked too. With these frequent accidents of cloud, the users start more consideration about personal privacy which would be able to get effective protection when the data is stored in the cloud. To ensure the confidentiality of data, a growing number of companies and individual users choose to encrypt data and allow the data in the form of ciphertext stored in the cloud server.

To solve this problem, Searchable Encryption is introduced [6-18]. Using SE mechanism encrypts data, and the ciphertext is stored in the cloud server. When users need to search some keywords, they can use the keyword to search documents sent to the cloud server.The cloud will receive the search proof test matching for each file, if the match is successful, it implies that the file contains the keyword. Finally, the cloud will return all files matching success back to the user. After receiving the search results, users only need to return to the encrypted files. The majority of the schemes study single keyword, conjunctive keywords and complex search query of public key cryptography based SE schemes [1], [19-23].

Received August 20, 2016; Revised October 12, 2016; Accepted October 28, 2016

Here is a motivating example for PEKS. This example is according to the reference [1]. Suppose user Alice wants to read her emails from her laptop or smartphone or PAD after she stores her emails in the servers of some email service provider. Because of previous cloud accidents, Alice does not believe the third-party service provider or fears that powerful agencies may require the service provider to surrender all her data. Any user with Alice's public key can send her encrypted emails from the many transmission mediums that only she can decrypt based on standard public key encryption. PEKS scheme produces some email searchable ciphertexts, Alice prepare to find a unique email then, the sender could also attach to the searchable ciphertexts. Alice could make use of keywords to search for this email. Once delegated, the ciphertexts can be searched. Across Alice's email the service provider searches those search ciphertexts contained that match the issued trapdoor, and returns to her a positive match.

Based on PEKS, there [24] has put forward the hidden ciphertext search mechanism of user access patterns firstly. Even if later proposed [25] asymmetric searchable encryption that had a disadvantage of low efficiency and leaking search pattern. Many of the articles study the primitives Asymmetric searchable encryption with more nimble search queries pattern, such as single keyword, multi-keyword, conjunctive keyword query and inner product types of queries. In recent years, some schemes still protect the ciphertexts's data privacy, but [26] about leaking search pattern has taken little attention. In this paper we introduce a new efficient project that it can distribute the search on the encrypted data to the storage provider and the query proxy. Therefore, we call our new construction a distributed Asymmetric Searchable Encryption (DSAE) scheme. The proposed DSAE scheme achieves its security and efficiency because of the distributed estimation and the use of complex search query like C3DH and pseudo-random functions only. In this way, the DSAE not only can hide search also prevent leakage data privacy of users and the more important is to improve the efficiency of searching way.

1.1. Related work

In 2004, Dan Boneh first introduced Public Key Encryption with keyword search to solve security problem. In 2005, Michel Abdalla discussed some extensions of searchable encryption based on identity-based encryption (IBE) and consistency-related issues. Ballard proposed SE mechanism based on public key cryptography, using the bilinear mapping to make the trapdoor fixed, but its security was established on DDH, XDH and MXDH complexity. In 2001, the paper's [29] work could make the server rank user's requested search multiple keywords, according to the score of each file for the requested keywords and return the highest ranked k files to the user, the server would not be able to acquire the search keywords information of users, if the file contains a keyword information and the final score of every file information.

In 2013, Qiang Tang [27] proposed Asymmetric Searchable Encryption with Message Recovery and Flexible Search Authorization when the owners of data used the primitive to keep their data encrypted and different search access server assigned to a third party which based on DLIN and BDH assumptions in the random oracle model and used bilinear pairings and designed double encryption attribute technology. At the same year, he put forward a scheme Extend the Concept of Public Key Encryption with Delegated Search[28] which contained the multiple-server setting and presented a new security model.

In 2014, Christoph Bosch and Qiang Tang firstly proposed Distributed Searchable Symmetric Encryption which uses a query proxy to achieve and security efficiency at the same time.

1.2. Our Contribution

In this paper, we provide a different efficient construction of DSAE scheme and provide a wider view on what can be comprehended regarding Distribute and asymmetric searchable encryption and integrate the distinction between currently existent definitions. Our contributions can be summarized as below:

(1) We introduce the polyfunctional concept of Searchable encryption which combines Distributed and Asymmetric.

(2) We put forward a new security model, strictly efficiency superior than Distributed Searchable Symmetric Encryption (DSSE) [30]. In the search query uses complex search query and the leaking of search way greatly reduces.

(3) We compare the distinct notions of security and efficiency between DSSE and DSAE which shows that privacy in DSAE is independent.

(4) In this paper, the proof of safety is based on Composite 3-party Diffie-Hellman (C3DH) assumption made in [31] by Boneh and Waters.

2. Preliminaries

2.1. PEKS Theory

Definition 1. We have known the formal and notion definition of PEKS (as defined in [1], [32]). In general, a public key encryption search system includes four probabilistic polynomial time algorithms as follows:

1) Setup (k): Enter a security parameters k, and output a public key Bkpr, a private key Bkpr as well as public key encryption system parameters S [14].
2) PEKS (Bkpr, W): Input a keyword W, a message M and a public key Bkpr, export a
3) searchable encryption of Bkpu, $B_{kpu} \leftarrow S_{B_{kpu},W}$
4) Trapdoor (Bkpr, W): Input a private key Bkpr and a message M, calculate the trapdoor value about W, $W \leftarrow T_W$
5) Test (Bkpu, P, TW): Input a searchable encryption P and a trapdoor value TW, P=PEKS (Bkpu,W'), TW=Trapdoor(Bkpr,W). If W=W' output M, otherwise output aborted.

Given the above definition, a performance of a public-key encryption scheme with keyword search goes as follows. First, the receiver uses the Setup algorithm to produce her public or private key pair. Then, she runs the Trapdoor algorithm to create trapdoors T_W for any keyword *W,* which the third service providers can search for. The given trapdoors are as input to the Test algorithm by third service providers to determine whether one sender gives message encrypted using algorithm *ksEnc* containing one of the keywords *W* specified by the receiver**.**

2.2. Bilinear groups

Definition 2. Nowadays the widely applications of searchable encryption system built on bilinear pairings which based on public key cryptography. Its security is based on different security assumptions. The following Table 1 gives the definition of the bilinear pairings firstly.

Table 1. Symbols of Bilinear Pairings

P	a λ - bit prime.
G, G_T	groups of order *p*
g	a generator of *G*
e	is an efficiently-computable bilinear pairing $e: G_1 \times G_2 \rightarrow G_2$
ε	output by $\varepsilon(1^k) \leq < p, G_1, G_2, \hat{e} >$

Bilinear. About bilinear mapping $e: G_1 \xi G_2 \rightarrow G_2$, need to satisfy the following conditions:

Bilinear: $\forall a, b \in \phi q, \forall g, h \in G_1, e\left(g^a, h^b\right) = e(g, h)^{ab}$;

Non-degenerate: $\exists g \in G_1$, make $e(g, g) \neq 1$;

Computable: compute of group G_1, G_2 and can be solved in polynomial time about bilinear mapping *e*.

2.3. Security assumptions

Definition 3. There exist a lot of security assumptions. Two typical revised assumptions are Decision Bilinear Diffie-Hellman (DBDH) and Decision Linear (DLA) assumptions [33, 34]. Then new Composite Decision Die-Hellman (CDDH) assumption has been proposed, and there has proof to show that this assumption is weaker than the well-established Composite 3-party Die-Hellman (C3DH) assumption made in [31].

Let ∂ be a DBDH parameter generator, ∂ output $\Gamma=(p,G_1,G_2,\hat{e},g)$, in Figure 1 for every *PPT* adversary *A*, DBDH assumption holds for description Γ [35], Given $<g,ag,bg,cg>$ for $a,b,c\in Z_p^*$ compute $W=\hat{e}(g,g)^{abc}\in G_2$.

If $\Pr(A(g,ag,bg,cg)=\hat{e}(g,g)^{abc})\geq\varepsilon$, *A* has advantage ε in $(G_1,G_2,\hat{e})$. We can conclude that *A* has advantage $\varepsilon(k)$ in solving the DBDH problem for *G* if for sufficiently large *k* (security parameter $k\in Z^+$):

$$Adv_{\Gamma,A(k)}^{DBDH}=\Pr\left[A\left(p,G_1,G_2,\hat{e},g,ag,bg,cg\right)=\hat{e}(g,g)^{abc}\mid \begin{array}{c}<p,G_1,G_2,\hat{e}>\leftarrow\vartheta(1^k),\\ g\leftarrow G_1^*,a,b,c\leftarrow Z_q^*\end{array}\right]\geq\varepsilon(k)$$

If for any randomized polynomial time (in *k*) algorithm *A* we have ∂ to meet the DBDH assumption. $Adv_{\Gamma,A(k)}^{DBDH}$ is a negligible function.

$$Adv_{\Gamma,A(k)}^{DBDH}:=2\bullet\Pr[DBDH\Rightarrow True]-1$$

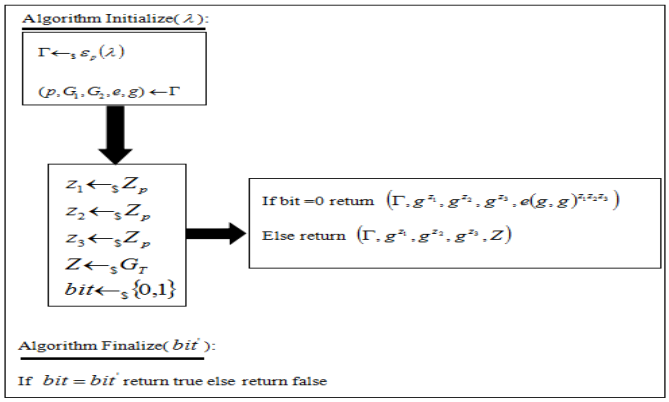

Figure 1. DBDH assumption

Definition 4. In the same way we can get DLA assumption (Figure 2). Let ∂ be a DLA parameter generator, ∂ output $\Gamma=(p,G_1,G_2,\hat{e},g)$, for every *PPT* adversary *A*, DLA assumption holds for description Γ. Given $<g,ag,bg,cg>$ for $a,b,c\in Z_p^*$ compute $W=\hat{e}(g,g)^{abc}\in G_2$.

If $\Pr(A(g,ag,bg,cg)=\hat{e}(g,g)^{abc})\geq\varepsilon$, *A* has advantage ε in $(G_1,G_2,\hat{e})$. We can conclude that *A* has advantage $\varepsilon(k)$ in solving the DLA problem for *G* if for sufficiently large *k*（security parameter $k\in Z^+$):

$$Adv_{\Gamma,A(k)}^{DLA}=\Pr\left[A\left(p,G_1,G_2,\hat{e},g,ag,bg,cg\right)=\hat{e}(g,g)^{abc}\mid \begin{array}{c}<p,G_1,G_2,\hat{e}>\leftarrow\vartheta(1^k),\\ g\leftarrow G_1^*,a,b,c\leftarrow Z_q^*\end{array}\right]\geq\varepsilon(k)$$

If for any randomized polynomial time (in *k*) algorithm *A* we have ∂ to meet the DBDH assumption. $Adv_{\Gamma,A(k)}^{DLA}$ is a negligible function.

$Adv_{\Gamma,A(k)}^{DLA} := 2 \cdot \Pr[DLA \Rightarrow True] - 1.$

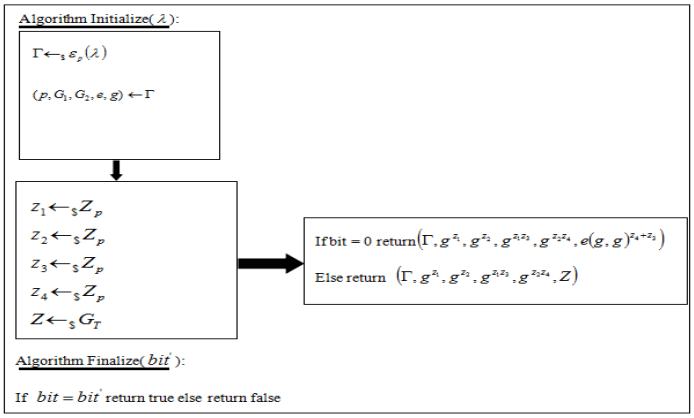

Figure 2. DLA Assumption

Definition 5. The Composite 3-Party Diffie-Hellman Assumption (C3DH) is a very important assumption about complex search query in public key cryptography. An algorithm ε_C needs to input a security parameter λ and output a description $\Gamma = (t, r, G, G_T, e, g)$, we need pay attention to the follows:

1) *G* and G_T are groups of order *n=tr*, where *t* and *r* are independent $\lambda - bit$ primes.
2) *g* is a generator of *G*.
3) *e* is an efficiently-computable bilinear pairing $e: G \times G = G_T$, a map meets two properties as follows:
   a. Bilinearity: $\forall a, b \in Z_n$, $e(g^a, g^b) = e(g,g)^{ab}$.
   b. Non-degeneracy: $e(g,g) \neq 1$.

For a given group generator Ψ, then define the $P(\lambda)$:

$$(t, r, G, G_T, e) \xleftarrow{R} \Psi(\lambda), n \leftarrow tr, g_t \xleftarrow{R} G_t, g_r \xleftarrow{R} G_r$$
$$R_1, R_2, R_3 \xleftarrow{R} G_t$$
$$a, b, c \xleftarrow{R} Z_n$$
$$\bar{Z} \leftarrow \left((n, G, G_T, e), g_r, g_t, g_t^a, g_t^b, g_t^{ab} R_1, g_t^{abc} R_2\right)$$
$$T \leftarrow g_t^c R_3$$

Output $(\bar{Z}, T)$

For an algorithm *B*, define B' is advantage in solving the composite 3-party Diffie-Hellman problem for Ψ as:

$$Adv_{\Psi,B(\lambda)}^{C3DH} := \left| \Pr\left[B\left(\bar{Z}, T\right) = 1 \right] - \Pr\left[B\left(\bar{Z}, R\right) = 1 \right] \right|$$

Where $(\overline{Z},T)\xleftarrow{R}P(\lambda)$ and $R\xleftarrow{R}G$.

Definition 5: Let $\Psi_C(\lambda)$ output the description $\Gamma=(t,r,G_T,e,g)(n\leftarrow tr)$. We say the Ψ satisfies the composite 3-party Diffie-Hellman assumption (C3DH) if for any polynomial time algorithm *B*. we have that the function $Adv^{C3DH}_{\Psi,B(\lambda)}$ is a negligible function of λ.

C3DH is described in Figure 3.

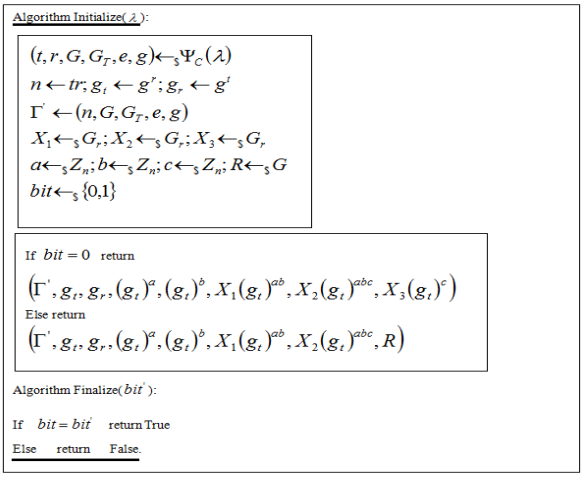

Figure 3. D3DH Assumption

3. The Proposed DSAE Construction

Through this paper, we must use the following notions.shown in Table 2.

Table 2. Some Notions

Symbol	Definition
$D=\{d_1,d_2,\ldots,d_n\}$	a set of *n* files
d	a document
u(d)	a set of different keywords
$W=\{w_1,w_2,\ldots,w_m\}$	a pre-built dictionary of m keywords
t	a tuple
t[i]	the *i*-th entry of *t*
$I=\{I_{w_1}\ldots\ldots I_{w_m}\}$	encrypted index
J	a re-encrypted index
I'	a re-encrypted and permuted index
$s\in W$	keyword used in a query
$a\xleftarrow{\$}A$	an element *a* randomly chosen from a set *A*
λ	a security parameter
p, q	$\lambda-bit$ prime
G,G_T	groups of order *p,q*
g	a generator of *G*
e	an efficiently-computable bilinear pairing $e:G\times G\rightarrow G_T$

DSAE scheme is a protocol between three parties (a client *C*, a storage provider *SP*, a query proxy *QP*). The scheme consists of 4-tuple of algorithms (KeyGen, Encrypt, Trapdoor, Test).

1) KeyGen: $(K_C,K_{SP_1},K_{SP_2},K_{QP_1},K_{QP_2})=Keygen(\lambda)$.This algorithm is executed by the

client *C*, inputs a security parameter λ and outputs a secret key K_C to the client *C*, public keys K_{SP_1} and K_{QP_1} to *SP* and *QP*($K_{SP_1} = K_{QP_1}$) and private keys K_{SP_2} and K_{QP_2} to *SP* and *QP*, respectively.

2) Encrypt: $(I_1, I_2) = Encrypt(K_C, D)$. This algorithm is executed by the client *C*, inputs a key K_C and a set of documents *D*, outputs an encrypted index I_1 to *SP* and I_2 to *QP*.
3) Trapdoor: $(T_1^s, T_2^s) = Trapdoor(K_{SP_2}, K_{QP_2}, s)$.This algorithm is executed by the client *C*, inputs the private key K_{SP_2} , K_{QP_2} and a query keyword $s \in W$, and outputs a trapdoor T_1^s to *SP* and trapdoor T_2^s to *QP*.
4) Test: $a = Test(K_{SP_1}, K_{QP_1}, I_1, I_2, T_1^s, T_2^s)$. *SP* provides K_{SP_1} , I_1 , T_1^s and *QP* provides K_{QP_1} , I_2 , T_2^s as input. According to the matching results of *W* and *W'* outputs judgment value *a*, $a \in \{0,1\}$.

In addition, we require a DSAE scheme to correct that for a set of keywords *W*, any set of documents *D*, all security parameter λ , all output

$$K_C, K_{SP_1}, K_{QP_1}, K_{SP_2}, K_{QP_1} \leftarrow keygen(\lambda),$$
$$(I_1, I_2) = Encrypt(K_C, D),$$
$$(T_1^s, T_2^s) = Trapdoor(K_{SP_2}, K_{QP_2}, s)$$

and all keyword $w, s \in W$, it holds that: $Test(K_{SP_1}, K_{QP_1}, I_1, I_2, T_1^s, T_2^s) = id_s(D)$; where $id_s(D)$ denotes the set of identifiers of all documents in *D* containing the keyword *s*. The sequence of document identifiers $id_s(D)$ for consecutive keywords *s* is called the access pattern.

Now, we make some assumptions that the *i-th* query queries for $s_i \in W$ when client makes *Q* query. So in total the client queries for $s_1, s_2 \ldots\ldots, s_Q \in W$. T^{s_i} expresses a trapdoor for the *i-th* query. We denote an admissible protocol run of a DSAE scheme, where the client performs *Q* queries, by $\prod_{DASE}^{Q}$. Formally, an admissible Q-query protocol run $\prod_{DASE}^{Q}$ which is defined as follows:

DSAE scheme set keyword *W*, output $(K_C, K_{SP_1}, K_{SP_2}, K_{QP_1}, K_{QP_2}) = Keygen(\lambda)$ and a document set *D*. An admissible Q-query protocol ($Q \in N$) run consists of one call of algorithm $(I_1, I_2) = Encrypt(K_C, D)$, followed by *Q* calls of algorithm $(T_1^s, T_2^s) = Trapdoor(K_{SP_2}, K_{QP_2}, s)$ for (possibly different) keywords $s_i \in W$ and $i \in [1, Q]$, and another *Q* calls of algorithm $Test(K_{SP_1}, K_{QP_1}, I_1, I_2, T_1^s, T_2^s)$. We denote such a protocol run by $\prod_{DASE}^{Q}$.

4. Security Model

SSE is considered a trusted party. For *SP* and *QP*, we suppose secure channels between the three parties which do not collude. It indicates that admissible *Q-query* protocol running $\prod_{DASE}^{Q}$ ($Q \in N$) are executed. Normally, to all participants, the protocol $\prod_{DASE}^{Q}$ has the unique public output access pattern ($id_{w_1}(D) \ldots\ldots id_{w_Q}(D)$). If a DSAE scheme is secure, it leaks no information. The following is that we first define ideal functionality of a DSAE scheme:

Functionality X_{DASE}^{Q}. Consider a DSAE scheme with keyword set *W*, output $(K_C, K_{SP_1}, K_{SP_2}, K_{QP_1}, K_{QP_2}) = Keygen(\lambda)$, and a document set *D*. X_{DASE}^{Q}($Q \in N$) is the functionality that takes as input:

1) K_C and keywords $w_1 \ldots\ldots w_Q$ from client *C*.
2) K_{SP_1} , K_{SP_2} from provider *SP*.
3) K_{QP_1} , K_{QP_2} from query proxy *QP*.
4) $id_{w_1}(D) \ldots\ldots id_{w_Q}(D) \rightarrow id_Q(D)$ to all *C*,*SP*,*QP*.

Then, we consider that a DSAE scheme is secure if all the admissible *Q-query* run $\prod_{DASE}^{Q}$ ($Q \in N$) compute functionality X_{DASE}^{Q}.We can use a formula (include a simulator *S*) to express as Figure 4.

$$SP:$$
$$\left\{S\left(K_{SP_1}, K_{SP_2}, id_Q(D)\right)\right\}_{K_C, w_1 \ldots\ldots w_Q, K_{SP_1}, K_{SP_2}, K_{QP_1}, K_{QP_2}}$$
$$\equiv_c \left\{View_{SP}\left(K_C, w_1 \ldots\ldots w_Q, K_{SP_1}, K_{SP_2}, K_{QP_1}, K_{QP_2}\right)\right\}_{K_C, w_1 \ldots\ldots w_Q, K_{SP_1}, K_{SP_2}, K_{QP_1}, K_{QP_2}}$$
$$QP:$$
$$\left\{S\left(K_{QP_1}, K_{QP_2}, id_Q(D)\right)\right\}_{K_C, w_1 \ldots\ldots w_Q, K_{SP_1}, K_{SP_2}, K_{QP_1}, K_{QP_2}}$$
$$\equiv_c \left\{View_{QP}\left(K_C, w_1 \ldots\ldots w_Q, K_{SP_1}, K_{SP_2}, K_{QP_1}, K_{QP_2}\right)\right\}_{K_C, w_1 \ldots\ldots w_Q, K_{SP_1}, K_{SP_2}, K_{QP_1}, K_{QP_2}}$$

It should note that it is enough to simulate the process of SP and QP separately. Because we do not consider any form of collusion between them. Remember that the client is regarded as a trusted party which only provides inputs. So the security definition of it does not need to take the client's view into consideration.

Figure 4. Security Model

4.1. Our Construction

In this part, we use the following symbols as Table 3.

Table 3. Symbols

Symbol	Definition
p_w	A bit string of length *n* for *w*
n	The number of the documents
j	A unique document identifier
$p_w[j]$	A unique document
$\oplus$	A bitwise XOR operation

We have known that a DSAE scheme consists of three parties: a client *C*, a storage provider *SP* and a query proxy *QP*. DSAE scheme uses an index per distinct keyword in the database. If $w \in d_j$, the *j-th* of p_w is set to 1. Otherwise, the bit is set to 0. Our construction makes use of the following cryptographic primitives (as Table 4).

We can construct new probability of polynomial.

1) $(K_C, K_{SP_1}, K_{SP_1}, K_{QP_1}, K_{QP_1}) = Keygen(\lambda)$. Input a security parameter λ, generate a key $K=(K_C=(K_f,k_P),K_1)$ for the pseudo-random functions. The key K_C is only known by *C*, the key K_{SP_2} is known by *C* and *SP*.
2) $(I_1, I_2) = Encrypt(K_C, D)$. Input the key K_C, and a document collection D.

3) $(T_1^w, T_2^w) = Trapdoor(K_{SP_2}, K_{QP_2}, s)$. Input the key K_{SP_2}, K_{QP_2} and a query keyword $s \in W$.

4) $a = Test(K_{SP_1}, K_{QP_1}, I_1, I_2, T_1^s, T_2^s)$. (*SP* provides I_1, T_1^s and *QP* provides I_2, T_2^s).

SP re-encrypts and permutes the index *I* for all $i \in [1,m]$ as follows:
$J = \{J_{w_i}\} = \{I_{w_i} \oplus g_{w_i}(r_1) \oplus h(r_1)\}$, $I' = \pi(J, p_{r_2})$ and sends T' to *QP*.

Table 4. Cryptographic Primitives

$f(K_C, w)$	The function inputs a key K_C and a keyword *w*, outputs a pseudo-random bit-string of length *n*.
$g(K_{SP_2}, w, r_1)$	The function inputs a key K_{SP_2}, a keyword *w* and a random value r_1. It outputs a pseudo-random bit-string of length n.
$h(K_{SP_1}, r_1)$	The function inputs a key K_{SP_2}, and a random value r_1. It outputs an *n-bit* pseudo-random string.
$g(K_{QP_2}, w, r_2)$	The function inputs a key K_{QP_2}, a keyword *w* and a random value r_2. It outputs a pseudo-random bit-string of length *n*.
$h(K_{QP_1}, r_2)$	The function inputs a key K_{QP_2}, and a random value r_2. It outputs an *n-bit* pseudo-random string.
p_k	The keyed pseudo-random permutation pk describes a permutation on the set $[1,m]$. The evaluation of the permutation p_k takes as input an element $x \in [1,m]$ and outputs its permuted position $p_k \in [1,m]$.
$\pi(X, p_x)$	The function inputs a set *X* of size $\|W\|$ and a random permutation p_x. It outputs a permuted set according to p_x.

5. Security analysis

$s_1 \ldots\ldots s_Q$ denote the client queries for the *Q* keywords. The query proxy *QP* learns the values $q_{s_j}(r_2)$ and *k* for every keyword s_j ($j \in (1,Q)$). Since $q_{s_j}(r_2)$ is an index position for keyword s_j after a pseudo-random permutation with function π with input *J* and the pseudo-random permutation based on the random value r_2, the value can be simulated by choosing a random value between 1 and *m*. The value *k* is computed as an XOR of the n-bit outputs of the pseudo-random functions $f(K_C, s_j)$ and $g(K_{SP2}, s_j, r_1)$ and the random n-bit string r_3. The value *k* is thus indistinguishable from random and can be simulated by *S* with random n-bit string. In total, this shows that *S* successfully simulates the view of the query proxy *QP* as Figure 5.

6. Performance analysis

In this section, we introduce the efficiency of our proposed DSAE scheme.The search cap size is *O(D)*. The encryption time is *O(W)G*. The search time is *O(KW)P*. This algorithm produces an n-bit string (f_s) for all keywords $s \in W$. Let a_1 , a_2 be the number of *SP*, *QP* keywords respectively and let *b* be the number of documents. So $a_1 \times a_2 \times b$ is a matrix of resulting index. This process generates $a_1 \times a_2$ times f_s and it will calculate $a_1 \times a_2 \times b$. In Encrypt algorithm, the size complexity is O($a_1 \times a_2 \times b$). The Trapdoor algorithm selects three random values $r_1 \leftarrow C$, $r_2 \leftarrow SP$, $r_3 \leftarrow QP$. It evaluates $\pi(W, p_{r_1})$, $\pi(W, p_{r_2})$ and $\pi(W, p_{r_3})$ of keyword *s*. Generating three n-bit strings ($f_s, g_s(r_1)$), ($f_s, g_s(r_2)$), ($f_s, g_s(r_3)$). So the Trapdoor size is *O(n)*. In Test algorithm, *SP* produces *m+1* n-bit strings and performs a random evolution on *m* index positions. The complexity is of Test is *O(mn)*.

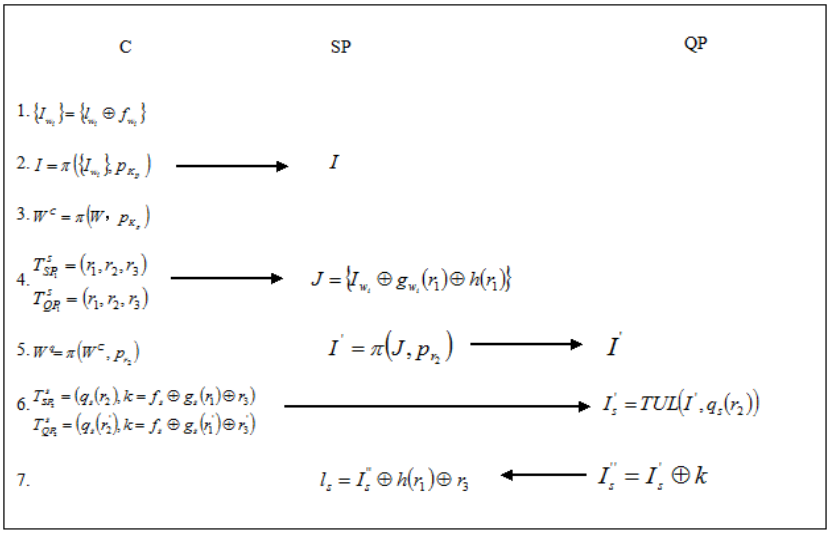

Figure 5. Simplified Upload and Search Processes of our DSAE Scheme

7. Conclusions

In this paper, we have proposed the concept of distributed searchable asymmetric encryption (DSAE) for outsourcing encrypted data. Compared with standard DSSE, a DSSE scheme can potentially provide more efficiency and better security guarantees. We described a security model and previous models also protect the search pattern. We proposed a construction for DSAE (based entirely on binary XOR operations and pseudo-random functions) which is highly efficient, despite the additional security. The scheme uses an inverted index approach and borrows re-shuffling techniques from private information retrieval. The main idea is that the query proxy gets a fresh (i.e., re-encrypted and shuffled) index per query. Thus, the query can be realized by a simple table look-up without revealing the search pattern. We also have shown that even if the storage provider and query proxy collude, the scheme is still secure under Curtmola's definition for adaptive semantic security of SSE. When the two servers collude, the resulting SSE scheme is very efficient and it outperforms Curtmola's scheme in terms of trapdoor sizes.

References

[1] Boneh D, Di Crescenzo G, Ostrovsky R, et al. Public key encryption with keyword search. Advances in Cryptology-Eurocrypt 2004. *Springer* Berlin Heidelberg. 2004; 506-522.
[2] Asonov D, Christoph J, Freytag JC. Private Information Retrieval. Jahrestagung. 2001; (2): 889-894.
[3] Cachin C, Micali S, Stadler M. Computationally Private Information Retrieval with Polylogarithmic Communication. Advances in Cryptology-EUROCRYPT '99. *Springer*. Berlin Heidelberg. 1999: 402-414.
[4] Chor B, Gilboa N, Naor M. *Private Information Retrieval by Keywords.* Ndss Symposium. 1997; 19(6): 535-540.
[5] Sculpher M. Using economic modelling to contribute to the prioritisation and design and clinical trials: ready for prime time. *Trials,* 2011; 12: 1-1.
[6] Song D, Wagner D, Perrig A. *Practical techniques for searches on encrypted data*. Proc. of the IEEE Symp on Security and Privacy. Berkeley: IEEE Computer Society. 2000; 44-55.
[7] Waters B, Balfanz D, Durfee G, Smetters D. *Building an encrypted and searchable audit log*. Proc. of the 11th Annual Network and Distributed System Security Symp. San Diego: The Internet Society, 2004.
[8] Goh E. Secure Indexes. In: Cryptology ePrint Archive. 2003.
[9] Golle P, Staddon J, Waters B. *Secure conjunctive keyword search over encrypted data*. In: Proc. of the 2nd Int'l Conf. on Applied Cryptography and Network Security (ACNS). Berlin, Heidelberg: Springer-Verlag. 2004; 31-45.
[10] Wang C, Cao N, Li J, Ren K, Lou WJ. *Secure ranked keyword search over encrypted cloud data*. In: Proc. of the IEEE 30th Int'l Conf. on Distributed Computing Systems (ICDCS). Genoa: IEEE Computer Society. 2010; 253-262.

[11] Li J, Wang Q, Wang C, Cao M, Ren K, Lou WJ. *Fuzzy keyword search over encrypted data in cloud computing.* Proc. of the IEEE INFOCOM Mini-Conf. San Diego: IEEE Computer Society. 1-5, 2010.
[12] Li M, Yu S, Cao N Lou W. *Authorized private keyword search over encrypted data in cloud computing.* In: Proc. of the IEEE Int'l Conf. on Distributed Computing Systems (ICDCS). Minneapolis: IEEE Computer Society. 2011: 383-392.
[13] Chang YC, Mitzenmacher M. *Privacy preserving keyword searches on remote encrypted data*. Proc. of the 3rd Int'l Conf. on Applied Cryptography and Network Security (ACNS). Berlin, Heidelberg: Springer-Verlag. 2005; 442-455.
[14] Boneh D, Crescenzo G, Ostrovsky R, Persiano G. Public key encryption with keyword search. In: Proc. of the EUROCRYPT. Berlin,Heidelberg: Springer-Verlag. 2004: 506-522.
[15] Shi E, Bethencourt J, Chan T, Song D, Perrig A. *Multi-Dimensional range query over encrypted data.* Proc. of the IEEE Symp. on Security and Privacy. Berkeley: IEEE Computer Society. 2007; 350-364.
[16] Shi E, Waters B. *Delegating capabilities in predicate encryption systems.* In: Proc. of the 35th Int'l Colloquium on Automata, Languages and Programming (ICALP). Berlin, Heidelberg: Springer-Verlag. 2008; 560-578.
[17] Yang Z, Zhong S, Wright R. *Privacy-Preserving queries on encrypted data.* In: Proc. of the 11th European Conf. on Research in Computer Security. Berlin, Heidelberg: Springer-Verlag. 2006; 479-495.
[18] Boneh D, Waters B. *Conjunctive, subset, and range queries on encrypted data*. In: Proc. of the 4th Conf. on Theory of Cryptography. Berlin, Heidelberg: Springer-Verlag. 2007: 535-554.
[19] Dong C, Russello G, Dulay N. *Shared and searchable encrypted data for untrusted servers.* In: Proc. of the 22nd Annual IFIP WG 11.3 Working Conf. on Data and Applications Security. Berlin, Heidelberg: Springer-Verlag. 2008: 127-143.
[20] Hwang Y, Lee P. *Public key encryption with conjunctive keyword search and its extension to a multi-user system.* In: Proc. of the Int'l Conf. on Pairing-Based Cryptography. Berlin, Heidelberg: Springer-Verlag. 2007: 2-22.
[21] Ballard J, Kamara S, Monrose F. *Achieving efficient conjunctive keyword searches over encrypted data.* In: Proc. of the 7th Int'l Conf. on Information and Communications Security. Berlin, Heidelberg: Springer-Verlag. 2005: 414-426.
[22] Okamoto T, Takashima W. *Hierarchical predicate encryption for inner-products.* In: Proc. of the ASIACRYPT. Berlin, Heidelberg: Springer-Verlag. 2009: 214-231.
[23] Katz J, Sahai A, Waters B. *Predicate encryption supporting disjunctions, polynomial equations, and inner products*. In: Proc. of the EUROCRYPT. Berlin, Heidelberg: Springer-Verlag. 2008: 146-162.
[24] Goldreich O, Ostrovsky R. Software protection and simulation on oblivious RAMs. Journal of the ACM. 1996; 43(3): 431-473.
[25] Qiang Tang. *Theory and Practice of Cryptography Solutions for Secure Information Systems, chapter Search in Encrypted Data*. Theoretical Models and Practical Applications. IGI. 2013; 84-108.
[26] Mototsugu Nishioka. Perfect keyword privacy in PEKS systems. *In Provable Security - ProvSec Springer*. 2012; 7496 of LNCS: 175-192.
[27] Tang Q, Chen X. *Towards asymmetric searchable encryption with message recovery and flexible search authorization.* Proceedings of the 8th ACM SIGSAC symposium on Information, computer and communications security. ACM. 2013: 253-264.
[28] Tang Q, Ma H, Chen X. Extend the Concept of Public Key Encryption with Delegated Search. *The Computer Journal*. 2013: bxt102.
[29] Cao N, Wang C, Li M, Ren K, Lou W. *Privacy-Preserving multi-keyword ranked search over encrypted cloud data*. Proc. of the IEEE INFOCOM. Shanghai: IEEE Computer Society. 2011: 829-837.
[30] Bösch C T, Peter A, Leenders B, et al. Distributed Searchable Symmetric Encryption. 2014.
[31] Dan Boneh and BrentWaters. Conjunctive, subset, and range queries on encrypted data. In Theory of Cryptography Conference (TCC). *Springer.* 2007; 4392 of LNC: 535-554.
[32] Abdalla M, Bellare M, Catalano D, Kiltz E, Kohno T, Lange T, Malone-Lee J, Neven G, Paillier P, Shi H. Searchable Encryption Revisited: Consistency Properties, Relation to Anonymous IBE, and Extensions. In: Shoup, V. (ed.) CRYPTO. *Springer*. Heidelberg. 2005; LNCS. 3621.
[33] Dan Boneh, Matt Franklin. Identity-based encryption from the Weil pairing. In Advances in Cryptology. *Springer.* 2001; 2139 of LNCS: 213-229.
[34] Shi E, Bethencourt J, Chan T, Song D, Perrig A. *Multi-Dimensional range query over encrypted data.* Proc. of the IEEE Symp. on Security and Privacy. Berkeley: IEEE Computer Society. 2007: 350-364.
[35] Arriaga A, Tang Q, Ryan P. Trapdoor Privacy in Asymmetric Searchable Encryption Schemes. Progress in Cryptology–AFRICACRYPT 2014. *Springer International Publishing*. 2014: 31-50.