Conjunctive Keyword Search With Designated Tester and Timing Enabled Proxy Re-Encryption Function for E-Health Clouds

Yang Yang and Maode Ma, *Senior Member, IEEE*

***Abstract*— An electronic health (e-health) record system is a novel application that will bring great convenience in healthcare. The privacy and security of the sensitive personal information are the major concerns of the users, which could hinder further development and widely adoption of the systems. The searchable encryption (SE) scheme is a technology to incorporate security protection and favorable operability functions together, which can play an important role in the e-health record system. In this paper, we introduce a novel cryptographic primitive named as conjunctive keyword search with designated tester and timing enabled proxy reencryption function (Re-dtPECK), which is a kind of a time-dependent SE scheme. It could enable patients to delegate partial access rights to others to operate search functions over their records in a limited time period. The length of the time period for the delegatee to search and decrypt the delegator's encrypted documents can be controlled. Moreover, the delegatee could be automatically deprived of the access and search authority after a specified period of effective time. It can also support the conjunctive keywords search and resist the keyword guessing attacks. By the solution, only the designated tester is able to test the existence of certain keywords. We formulate a system model and a security model for the proposed Re-dtPECK scheme to show that it is an efficient scheme proved secure in the standard model. The comparison and extensive simulations demonstrate that it has a low computation and storage overhead.**

***Index Terms*— Searchable encryption, time control, conjunctive keywords, designated tester, e-health, resist offline keyword guessing attack.**

I. Introduction

THE ELECTRONIC heath records (EHR) system will make medical records to be computerized with the ability to prevent medical errors [1]. It will facilitate a patient to create his own health information in one hospital and manage or share the information with others in other hospitals. Many practical patient-centric EHR systems have been implemented such as Microsoft Health Vault [2] and Google Health [3]. Given the ambitious prospect to deploy the EHR system ubiquitously, privacy concerns of the patients come up. Healthcare data collected in a data center may contain private information and vulnerable to potential leakage and disclosure to the individuals or companies who may make profits from them. Even though the service provider can convince the patients to believe that the privacy information will be safekeeping, the EHR could be exposed if the server is intruded or an inside staff misbehaves. The serious privacy and security concerns are the overriding obstacle that stands in the way of wide adoption of the systems.

Public key encryption scheme with keyword search (PEKS) [4]–[7] allows a user to search on encrypted information without decrypting it, which is suitable to enhance the security of EHR systems. In some situations, a patient may want to act as a delegator to delegate his search right to a delegatee, who can be his doctor, without revealing his own private key. The proxy re-encryption (PRE) method can be introduced to fulfill the requirement. The server could convert the encrypted index of the patient into a re-encrypted form which can be searched by the delegatee. However, another problem arises when the access right is disseminated. When the patient recovers and leaves the hospital or is transferred to another hospital, he does not want the private data to be searched and used by his previous physicians anymore. A possible approach to solve this problem is to re-encrypt all his data with a new key, which will bring a much higher cost. It will be more troublesome to revoke the delegation right in a scalable size.

In this paper, we endeavor to solve the problem with a novel mechanism proposed to automatically revoke the delegation right after a period of time designated by the data owner previously. In the traditional time-release system [28], [30], the time seal is encapsulated in the ciphertext at the very beginning of the encryption algorithm. It implies that all users including data owner are constrained by the time period. The beauty of the proposed system is that there is no time limitation for the data owner because the time information is embedded in the re-encryption phase. The data owner is capable to preset diverse effective access time periods for different users when he appoints his delegation right. An effective time period set by the data owner can be expressed with a beginning and closing time (for instance, 01/01/2014-12/01/ 2014). A time server is used in the system, which is responsible to generate a time token for the users. After receiving an effective time period T from the data owner, the time server generates a time

Manuscript received May 30, 2015; revised October 17, 2015 and December 1, 2015; accepted December 2, 2015. Date of publication December 17, 2015; date of current version February 1, 2016. This work was supported by the National Natural Science Foundation of China under Grant 61402112, Grant 61472307, and Grant 61472309. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Wanlei Zhou. *(Corresponding author: Yang Yang.)*

Y. Yang is with the School of Mathematics and Computer Science, Fuzhou University, Fujian 350108, China (e-mail: yang.yang.research@gmail.com).

M. Ma is with School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore 639798 (e-mail: emdma@ntu.edu.sg).

Digital Object Identifier 10.1109/TIFS.2015.2509912

seal S_T by using his own private key and the public key of the delegatee. In that way, the time period T is encapsulated in the time seal S_T. By the re-encryption algorithm executed by the proxy server, the time period T will be embedded in the re-encrypted ciphertext. It is the timing enabled proxy re-encryption function. When the delegatee issues a query request, he should generate a trapdoor for the queried keywords using his private key and time seal S_T. Only if the time period encapsulated in the trapdoor matches with the effective time period embedded in the proxy re-encrypted ciphertext, the cloud service provider will respond to the search query. Otherwise, the search request will be rejected. In that way, the access right of the delegatee will expire automatically. The data owner needs not to do any other operation for the delegation revocation.

To the best of our knowledge, this is the first work that enables automatic delegation revoking based on timing in a searchable encryption system. A conjunctive keyword search scheme with designated tester and timing enabled proxy re-encryption function (Re-dtPECK) is proposed, which has the following merits.

- 1) We design a novel searchable encryption scheme supporting secure conjunctive keyword search and authorized delegation function. Compared with existing schemes, this work can achieve timing enabled proxy re-encryption with effective delegation revocation.
- 2) Owner-enforced delegation timing preset is enabled. Distinct access time period can be predefined for different delegatee.
- 3) The proposed scheme is formally proved secure against chosen-keyword chosen-time attack. Furthermore, off-line keyword guessing attacks can be resisted too. The test algorithm could not function without data server's private key. Eavesdroppers could not succeed in guessing keywords by the test algorithm.
- 4) The security of the scheme works based on the standard model rather than random oracle model. This is the first primitive that supports above functions and is built in the standard model.

II. Related Work

A. Conjunctive Keyword Search

Various constructions of public key encryption with conjunctive keyword search (PECK) over encrypted data have been proposed [8]–[10]. It allows the users to query multiple keywords at the same time [11], [12]. However, some of them such as the solution in [9] and [10] have high communication or computation cost. On the other hand, some schemes such as the solutions in [8] and [12] require an index list of the queried keywords when a trapdoor is generated, which will leak information and impair the query privacy.

B. Searchable Encryption With Designated Tester

In practice, the size of a keyword space is always no more than its polynomial level. An attacker is possibly to launch dictionary attacks or off-line keyword guessing attacks (KG attacks) to exploit the hidden keywords. The EHR keywords

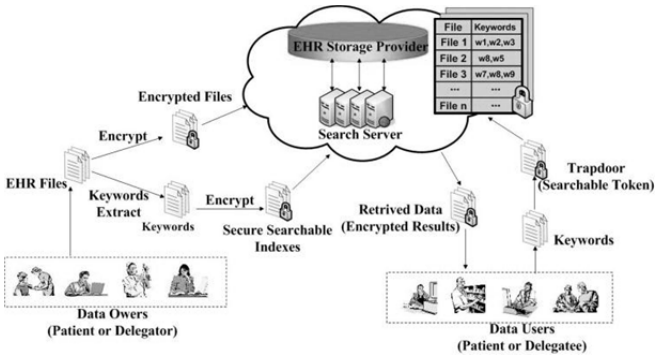

Fig. 1. System Model.

are usually selected from a small space, especially the medical terminology. If an adversary finds that the trapdoors or encrypted indexes have lower entropies, the KG attacks could be launched if the adversary endeavors to guess the possible candidate keywords. Byun *et al.* [19] and Yau *et al.* [20] have broken several classical schemes by the KG attacks.

In order to resist the threats, the concept of PEKS with designated tester (dPEKS) is proposed in [21]–[25]. Only a designated tester, which is usually the server, is capable to carry on the test algorithm. The enhanced security models [7], [26], [27] have also been put forward. However, they could not support multiple keywords query or delegate search function.

C. Proxy Re-Encryption With Public Keyword Search

Proxy re-encryption (PRE) enables a proxy with a re-encryption key to convert a ciphertext encrypted by a delegator's public key into those that can be decrypted by delegatee's private key. Proxy re-encryption with public keyword search (Re-PEKS) [13]–[15] has introduced the notion of keyword search into PRE. The users with a keyword trapdoor can search the ciphertext while the hidden keywords are unknown to the proxy. The limitation on the schemes in [13]–[15] is that only one keyword will be allowed to search in the encrypted documents. Later, Wang *et al.* [16] has suggested an improved scheme to support the conjunctive keyword search function. All these Re-PEKS schemes in [13]–[16] are proved secure in random oracle model. Nevertheless, it is shown in [17] and [18] that a proof in random oracle model may probably bring about insecure schemes.

The time controlled PRE has been addressed in [28]–[30]. It desires to encrypt a message for multiple recipients with the same release time. However, the schemes in [28] and [30] foist the data owner to determine the release time at the beginning of encryption algorithm. Only one release time is set for all recipients rather than disparate time for different users, which could not fulfill the need for uniqueness. Another shortcoming is that it needs a large computation cost in both encryption and re-encryption phases [29].

III. Problem Formulation

A. System Model

In Fig. 1, the environment of the proposed Re-dtPECK scheme for the EHR cloud system is presented. There are

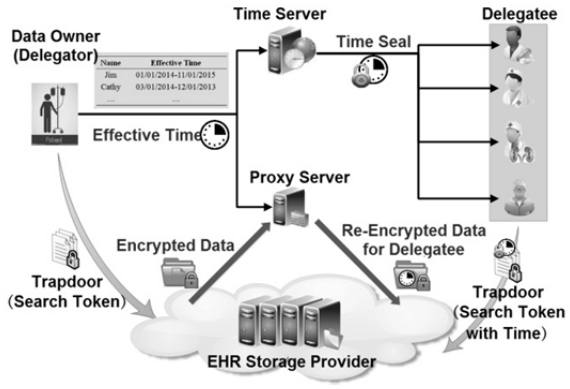

Fig. 2. Timing Enabled Proxy Re-encryption Searchable Encryption Model.

three types of entities: an information owner, users and a data center. The data owner wants to store his private EHR files on a third-party database. He extracts keywords from the EHR files and encrypts those plaintext keywords into the secure searchable indices. The EHR files are encrypted to ciphertext. Then, those information are outsourced to the data center.

A data center consists of an EHR storage provider and a search server. The storage provider is responsible for storing data and search server performs search/add/delete operations according to users' requests. A user generates a trapdoor to search the EHR files using his private key and sends it to the search servers. After receiving the request, the search servers interact with the EHR storage provider to find the matched files and returns those retrieved information to the user in an encrypted form.

In Fig. 2, the timing enabled proxy re-encryption searchable encryption model is shown. In this model, we highlight the implementation of the time controlled function. The data owner acting as a delegator sends a list of delegation effective time periods for his delegatees to the time server and the proxy server. The entry of the list contains the identity of each delegatee and the effective time period, such as "Jim, 01/01/2014 – 11/01/2015". It indicates that the delegatee Jim is authorized to issue queries and perform decryption operations on the encrypted data of the data owner from Jan. 1th, 2014 to Nov. 1th, 2015. After receiving the list, the time server generates a time seal for each delegatee, which is transmitted to individuals. The time seal is a trapdoor of an effective time period and concealed by the private key of the time server. In the re-encryption operation, the proxy server will encapsulate the effective time into the re-encrypted ciphertext. In order to reduce computing cost, the proxy server will not re-encrypt the ciphertext until they are accessed, which is so called lazy re-encryption mechanism [31]. In the query phase, the data owner can conduct ordinary search operations with his own private key. However, the delegatee has to generate a keywords trapdoor with the help of the time seal. The cloud data server will not return the matched files unless the effective time encapsulated in the time seal accords with the time in the re-encrypted ciphertext, which is different from traditional proxy re-encryption SE schemes.

B. Threat Model

The EHR data server is deemed as semi-trusted, who is honest to search information for the benefits of users but curious to spy out the private information of the patients. On the other hand, malicious outside attacker could eavesdrop and analyze the information transferred in public channel, such as the encrypted indexes and trapdoors. He intends to infer privacy information according to these data. Furthermore, the revoked delegatees may try to access data beyond the designated time period using their private keys. As most of the storage and search work are completed by the data server, it is assumed that the data server will not collude with the malicious outside attacker or revoked delegatees.

C. Design Goals

Our Re-dtPECK scheme for EHR cloud is designed to achieve the following goals.

- *1)* ***Authority delegation***. The proposed SE scheme should allow data-owner-enforced authority delegation, i.e. the data owner could delegate his search right to other users without revealing his private key.
- *2)* ***Time controlled revocation***. An important design goal is to enable time controlled access right revocation. The delegation appointment will terminate when the preset effective time period disagrees with the current time. It should prevent the authorized user from accessing the records overtime.
- *3)* ***Diverse delegation times for different users***. Another challenge of the system is to achieve owner-defined disparate access time periods for different delegatees. The data owner himself will not be constrained by the time.
- *4)* ***Security goals***. The privacy concerns of this secure search system are summarized as follows. 1) *keyword semantic security*: as a Re-dtPECK scheme is proposed, we will prove it indistinguishable against chosen keywords chosen time attack (IND-CKCTA). 2) *resist KG attacks*: since the EHR keywords are always chosen from a small space, the related searchable encryption schemes maybe vulnerable to offline KG attacks. The proposed scheme should resist such attack. 3) *standard model*: it is well known that security proved in standard model is stronger than that in random oracle model. This security property guarantees a higher security level.

IV. The Proposed Re-dtPECK

In this subsection, we formally define the conjunctive keyword search with a designated tester and the timing enabled proxy re-encryption function (Re-dtPECK). Then, we describe a concrete Re-dtPECK scheme with a detailed workflow and derive the correctness of the scheme. The notations presented in this paper are summarized in Table I.

A. Definition of Re-dtPECK

The Re-dtPECK scheme consists of following algorithms with an indicator θ. When its value is 1, the delegation function will be activated. Otherwise, the proxy re-encryption will not be enabled.

TABLE I
SUMMARY OF NOTATIONS

Notation	Description
θ	Delegation indicator
GP	Global parameter
sk_S, pk_S	Private and public keys of data server
sk_{TS}, pk_{TS}	Private and public keys of time server
sk_R, pk_R	Private and public keys of user
R_i, R_j	Delegator and Delegatee
W, Q	Keyword set
S_T	Time seal: trapdoor of effective time
$C_I (C_J)$	Ciphertext for delegator (delegatee)
$T_{Q,I} (T_{Q,J})$	Trapdoor of delegator (delegatee) on Q

- *GlobalSetup*(k): Taking a security parameter k as an input, this function generates a global parameter GP.
- $KeyGen_{Ser}(GP)$: Taking GP as an input, this algorithm generates a private and public key pair (sk_S, pk_S) for the data server.
- $KeyGen_{Rec}(GP)$: Taking a global parameter GP as an input, this function generates a private and public key pair (sk_R, pk_R) for the receiver.
- $KeyGen_{TS}(GP)$: Taking a global parameter GP as an input, this function generates a private and public key pair (sk_{TS}, pk_{TS}) for the time server.
- $dPECK(GP, pk_S, pk_{R_i}, sk_{R_i}, W)$: Taking GP, pk_S, pk_{R_i}, sk_{R_i} and a keyword set $W = (w_1, \cdots, w_l)$ as the inputs, the function returns a ciphertext C_I of W for R_i.
- $Trapdoor(GP, pk_S, sk_{R_i}, Q)$: Taking GP, pk_S, sk_{R_i} and a keyword query for $Q = (w_1, \cdots, w_m)$, $m \leq l$as the inputs, it outputs a trapdoor $T_{Q,I}$ for Q generated by R_i.
- $Test(GP, T_{Q,I}, sk_S, C_I)$: Taking GP, $T_{Q,I}$, sk_S and a ciphertext C_I of Was the inputs, the function returns '1' if W includes Q and '0' otherwise.

If the delegation indicator θ equals to 1, the following operations are executed.

- $ReKeyGen(GP, sk_{R_i}, sk_{R_j})$: Taking GP, sk_{R_i}, sk_{R_j} as the inputs, the algorithm outputs re-encryption key $rk_{R_i \to R_j}$.
- $Re-dtPECK(GP, rk_{R_i \to R_j}, C_I, pk_i, pk_j, pk_{TS}, T)$: Taking $GP, rk_{R_i \to R_j}, C_I, pk_i, pk_j, pk_{TS}, T$ as the inputs, the algorithm outputs a re-encryption ciphertext C_J.
- $TimeSeal(GP, sk_{TS}, T, pk_{R_i}, pk_{R_j})$: Taking GP, sk_{TS}, T, pk_{R_i}, pk_{R_j} as the inputs, it outputs a time seal S_T for user R_j in order to search user R_i's encrypted data.
- $Trapdoor_R(GP, pk_S, sk_{R_j}, Q, S_T)$: Taking GP, pk_S, sk_{R_j}, Q, S_T as the inputs, it outputs a trapdoor $T_{Q,J}$ for R_j.
- $Test_R(GP, T_{Q,J}, sk_S, C_J)$: Taking GP, $T_{Q,J}$, sk_S and C_J as inputs, the function returns '1' if W includes Q and the effective time contained in $T_{Q,J}$ is accordance with the time encapsulated in C_J. Otherwise, it outputs '0'.

Correctness: For any security parameter k, we have

$$Test \to \begin{cases} 1, & W \subseteq Q \\ 0, & otherwise \end{cases}, \quad Test_R \to \begin{cases} 1, & W \subseteq Q, T_1 = T_2 \\ 0, & otherwise \end{cases}$$

where T_1 is the effective time contained in $T_{Q,J}$ and T_2 is the effective time encapsulated in C_J.

The semantic security model is defined in Appendix. A.

B. Construction of Re-dtPECK

In the system, the EHR documents of the patients are encrypted by a symmetric encryption algorithm and the symmetric key is encapsulated with the patient's public key pk_A by the key encapsulation mechanism. The algorithms in the following focus on the searchable keywords encryption and the timing controlled delegation function. The time T chosen from the time space $\mathcal{T} \in \{0,1\}^{\varpi}$ has the form "Month/Day/Year-Month/Day/Year" to set the starting time and closing time of the delegation. For example, "05/01/ 2013-12/01/2014" means that the delegatee is authorized to operate on delegator's secret data from May 1st, 2013 to Dec. 1st, 2014. In this proposal, a hash function H is defined as $H : \{0,1\}^* \to Z_p^*$.

- *GlobalSetup*(k): Let k be the security parameter. Choose a random generator g of group G. The global parameter is $GP = g$.
- $KeyGen_{Ser}(GP)$ This algorithm randomly chooses $V \in G$ and $x \in Z_p^*$, computes $X = g^x$and outputs the data server's key pair $pk_S = (V, X)$, $sk_S = x$.
- $KeyGen_{Rec}(GP)$: For user R_i, choose a random value $y_i \in Z_p^*$ and compute $Y_i = g^{y_i}$. The public and private key pair of R_i is $pk_{R_i} = Y_i$ and $sk_{R_i} = y_i$.
- $KeyGen_{TS}(GP)$: This algorithm selects $\tau \in Z_p^*$ randomly and outputs a public and secret key pair for the time server as $pk_{TS} = g^{\tau}, sk_{TS} = \tau$.
- $dPECK(GP, pk_S, pk_{R_i}, sk_{R_i}, W)$: The sender selects a keyword set $W = (w_1, \cdots, w_l)$ from the file to be outsourced. An l-degree polynomial $\Upsilon(x) = \eta_l x^l + \eta_{l-1} x^{l-1} + \cdots + \eta_1 x + \eta_0$ should be constructed to make $y_i H(w_1), \cdots, y_i H(w_l)$ to be l roots of the equation $\Upsilon(x) = 1$. Then data sender randomly chooses $s, r \in Z_p^*$ and computes $C_1 = t \cdot e(g,g)^{-r}, C_2 = g^s, B_\psi = g^{r \cdot \eta_\psi}$, for $0 \leq \psi \leq l$, where $t = e(X,V)^s$. It then outputs $C_I = (C_1, C_2, B_0, B_1, \cdots, B_l)$. It is clear that the bilinear pairing operations $e(g,g)$ and $e(X,V)$ can be precomputed so that no additional time will be consumed.
- $Trapdoor(GP, pk_S, sk_{R_i}, Q)$: Select a random $T_{Q,-1}$, $\varsigma \in Z_p^*$ and compute $T_{Q,-2} = g^{\varsigma}, T_{Q,\psi} = g^{m^{-1} \cdot T_{Q,-1} \cdot (y_i)^{\psi} \cdot \sum_{\mu=1}^{m} H(w_{\gamma_\mu})^{\psi}} \cdot X^{\varsigma}$ where $0 \leq \psi \leq l$, $Qw = w(w_{\gamma_1}, \cdots, w_{\gamma_m})$, $m \leq l$. Then, the algorithm outputs the trapdoor $T_{Q,I} = (T_{Q,-1}, T_{Q,-2}, T_{Q,\psi})$ for $0 \leq \psi \leq l$.
- $Test(GP, T_{Q,I}, sk_S, C_I)$: After receiving the trapdoor $T_{Q,I}$ and the ciphertext C_I, the server computes $t = e(C_2, V)^x$ and checks whether the following equation holds $C_1^{T_{Q,-1}} \prod_{\psi=0}^{l} e[B_\psi, T_{Q,\psi}/(T_{Q,-2})^x] = t^{T_{Q,-1}}$. If the equation holds, it outputs 1. Otherwise, it outputs 0.

If delegation indicator θ equals to 1, execute the following operations.

The delegator R_i sends out a delegation notice to the trusted third party, time server, proxy server, data server and delegatee R_j. The delegation information is $\langle R_i, R_j, T, Sig_{sk_{R_i}}(R_i, R_j, T)\rangle$, in which T denotes the designated delegation time period and $Sig_{sk_{R_i}}(R_i, R_j, T)$ is a signature of the information (R_i, R_j, T) generated by delegator R_i used to verify the authenticity. *Sig* is a cryptographic secure signature algorithm without a specification in this scheme. The signature can be verified by using the public key of R_i. The delegation request could be rejected if the signature is forged. Otherwise, the following algorithms will be executed.

- $ReKeyGen(GP, sk_{R_i}, sk_{R_j})$: Taking GP, $sk_{R_i} = y_i$, $sk_{R_j} = y_j$ as the inputs, this algorithm outputs re-encryption key $rk_{R_i \to R_j} = y_i/y_j$.
- $Re - dtPECK(GP, rk_{R_i \to R_j}, C_I, pk_i, pk_j, pk_{TS}, T)$: This algorithm computes $C_3 = (g^{-H(T, pk_i)} pk_{TS})^{\mu}$, $C_4 = Y_j^{\mu}$, $C_5 = g^{\mu}$, $B'_{\psi} = B_{\psi}^{(rk_{R_i \to R_j})^{\psi}}$, for $0 \le \psi \le l$ and outputs a Re-dtPECK ciphertext $C_J = (C_1, C_2, C_3, C_4, C_5, B'_0, B'_1, \cdots, B'_l)$ for delegatee R_j.
- $TimeSeal(GP, sk_{TS}, T, pk_{R_i}, pk_{R_j})$: Taking the time server's private key $sk_{TS} = \tau$, the public key pk_{R_i}, pk_{R_j} and a designated time period $T \in \mathcal{T}$ as the inputs, it returns a time seal $S_T = (r_T, h_T)$, where r_T is a random element from Z_p^* and $h_T = (Y_j \cdot g^{-r_T})^{1/[\tau - H(T, pk_i)]}$.
- $Trapdoor_R(GP, pk_S, sk_{R_j}, Q, S_T)$: It computes $T_{Q,-1} = r_T, T_{Q,-2} = g^{\varsigma}, T_{Q,-3} = (h_T)^{y_j}, T_{Q,\psi} = g^{m^{-1} \cdot T_{Q,-1}(y_j)^{\psi} \sum_{\mu=1}^{m} H(w_{\gamma\mu})^{\psi}} \cdot X^{\varsigma}$, where $\varsigma \in Z_p^*, 0 \le \psi \le l, Q = (w_{\gamma_1}, \cdots, w_{\gamma_m}), m \le l$. Then, the output is $T_{Q,J} = w(T_{Q,-1}, T_{Q,-2}, T_{Q,-3}, T_{Q,\psi})$ for $0 \le \psi \le l$.
- $Test_R(GP, T_{Q,J}, sk_S, C_J)$: After receiving the search query from delegatee, the server firstly verifies whether the current time is within the delegation time period. If it is outside the delegation period, the request will be rejected. Otherwise, data server computes $t = e(C_2, V)^x$ and checks whether the following equation holds

$$C_1^{T_{Q,-1}} e(T_{Q,-3}, C_3) \prod_{\psi=0}^{l} e(B'_{\psi}, T_{Q,\psi}/(T_{Q,-2})^x) = t^{T_{Q,-1}} \cdot e\left(Y_j, C_4 \cdot C_5^{-T_{Q,-1}}\right).$$

If the equation holds, it outputs 1. Otherwise, it outputs 0. Appendix.C demonstrates the correctness of the scheme.

The designs goals in section III.C can be fulfilled based on the following aspects. 1) The authority delegation is realized mainly by proxy re-encryption mechanism. The proxy server makes use of the re-encryption key to transform the ciphertext encrypted by delegator's public key into another form, which can be searched by the delegatee using his own private key. 2) The delegatee will be divested of the search authority when the effective time expires. In order to achieve the time controlled access right revocation, the predefined time information is embedded in the re-encrypted ciphertext with a time seal. With the help of the time seal, the delegatee is able to generate a valid delegation trapdoor by $Trapdoor_R$ algorithm. If the time information hidden in the re-encrypted ciphertext is inconsistent with that in the delegation trapdoor, the equation in $Test_R$ algorithm will not hold. Moreover, the search query of the delegatee will be rejected by the data server if the current time beyond the preset time. 3) Since the time server is able to generate different time seals according to the requirement of the data owner, different delegation time periods can be set for different users through distributing the distinct time seals. The patient himself will not be constrained by the effective time period because the limitation is made in the delegation phase rather than the original encryption phase. 4) The security of this scheme is analyzed in Section V.

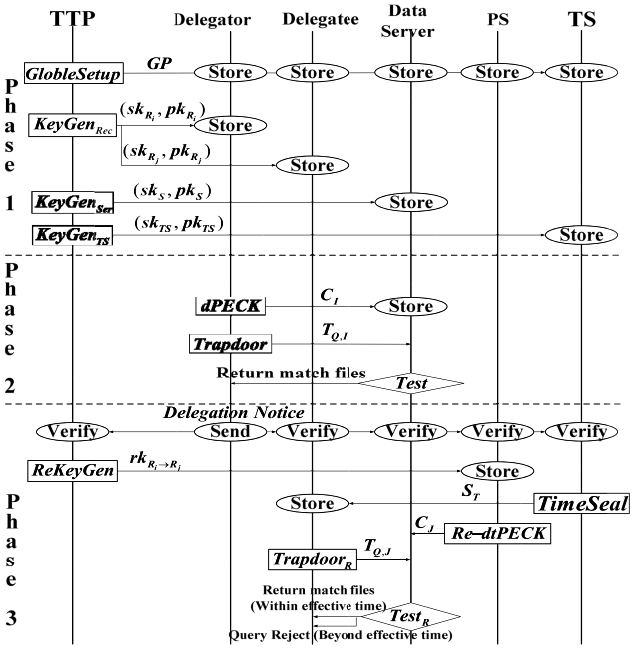

Fig. 3. Workflow of Re-dtPECK.

C. Workflow of Re-dtPECK

The detailed workflow is shown in Fig. 3 to illustrate the working procedure of the Re-dtPECK system in the EHR cloud. There are six entities to participate in the interactive process including a trusted third party (TTP). For instance, the Veterans Health Administration (VHA) is assumed to work as a TTP, who is trusted by clinics, hospitals, patients and doctors. A delegator is supposed to be Joe, who is a chronic heart failure patient. The EHR files of Joe are stored on a data server in the cloud in a protected form. Joe went to Hospital A for the cardiac treatment since Feb. 1st, 2014. He desires to designate the cardiologist Dr. Donne from Hospital A to be his delegatee for convenient EHR data access. Since Joe plans to transfer to Hospital B after June 1st and he hopes that Dr. Donne is not able to inquiry his EHR after that time. Then, Dr. Donne is granted a time-constrained authority to access the protected health information (PHI) of the patient Joe. The time server (TS) will generate a time seal for Dr. Donne to ensure that he is able to access to Joe's PHI during the period of Feb. 1st- May, 30st, 2014. The proxy server (PS) is responsible to encrypt Joe's PHI to a re-encrypted form so that Dr. Donne can search on those records with his own private key. The Re-dtPECK system can be divided into three phases.

In phase 1, the TTP initializes the system by executing *GlobalSetup* algorithm and generates the global parameters,

which are disseminated to delegator Joe, delegatee Dr. Donne, the EHR cloud server, the PS and the TS. The TTP also generates pairs of private and public key for Joe, Dr. Donne, the cloud server and the TS by running $KeyGen_{Rec}$, $KeyGen_{Ser}$, $KeyGen_{TS}$ algorithms.

In phase 2, EHR files are produced during Joe's therapeutic process. The encrypted EHR indices and documents will be generated using the dPECK algorithm and stored at the cloud data server. As the medical data accumulated, Joe may intend to search on his encrypted EHR files. He uses a keyword set Q to describe the health record file that he wants to find. Then, he runs *Trapdoor* algorithm to generate a trapdoor for keyword set Q and sends the trapdoor to cloud server. After receiving the search query, the cloud server runs test algorithm with the cloud server's private key and returns all the files that contain Q.

If the delegation indicator θ equals to 1, phase 3 will be executed. Joe sends a delegation notice to the TTP, PS, TS, delegatee and data server together with a signature signed by Joe. The effective delegation time of PHI access delegation for delegatee is specified. It means that the patient Joe has delegated the access rights to Dr. Donne. The recipients will verify the signature using the public key of Joe. In this system, the signature algorithm will not be specified. But there is a requirement on the algorithm that the signature scheme should be strongly unforgeable. The notice will be rejected if the signature fails the verification. If it is verified true, the TTP runs ReKeyGen algorithm to generate a re-encryption key and send it to the PS secretly. The TS runs *TimeSeal* algorithm to generate a time seal for delegatee. When Joe's PHI data is accessed by the Dr. Donne, the PS will run Re-dtPECK algorithm to encapsulate the effective time period into re-encrypted ciphertext. If the current time is not in accordance with the effective time period, the PS will not do the re-encryption operation for Dr. Donne. Suppose Dr. Donne is responsible to perform a heart bypass operation for Joe on Mar. 12nd, 2014. It is necessary for Dr. Donne to gain access to Joe's historical health record in order to prepare for the operation. Dr. Donne runs delegation trapdoor generation algorithm to get a valid trapdoor, which is used to conduct search query on Joe's PHI data. After receiving the query, cloud server runs the delegation test algorithm. The search query will be rejected if the current time beyond the effective time period. If the delegation test algorithm holds and current time is within the effective time period, the matched files will be returned to Dr. Donne.

V. Security Analysis

A. Hardness Problem

The following hardness problems are used in the security proof in next subsection.

Definition 1 (Truncated Decisional l -Augmented Bilinear Diffie-Hellman Exponent Problem (Truncated Decisional l-ABDHE)): Given a group G of prime order q, the algorithm randomly chooses generators g, g' from G and $\alpha \in Z_q^*$ at random. Given a tuple $Tu = (g', g'_1, g'_2, g'_{l+2}, g, g_1, \cdots, g_l)$ and $Z \in G_1$, where we use g_i and g'_i to denote g^{α^i} and $(g')^{\alpha^i}$, the truncated decisional l-ABDHE problem is to decide whether Z equals to $e(g', g_{l+1})$ or to a random element of G_1.

Definition 2 (Decisional Bilinear Diffie-Hellman Problem (DBDH)): Given a group G of prime order q, the algorithm randomly chooses generators g from G and $a, b, c \in Z_q^*$ randomly. Given a tuple $Tu = (g^a, g^b, g^c)$ and $Z \in G_1$, the DBDH problem is to decide whether Z equals to $e(g, g)^{abc}$ or to a random element of G_1.

Definition 3 (Decisional Diffie-Hellman Problem (DDH)): Given a group G of prime order q, the algorithm randomly chooses generators g from G and $\alpha, \beta \in Z_q^*$ at random. Given a tuple $Tu = (g, g^\alpha, g^\beta)$ and $Z \in G$, the DDH problem is to decide whether Z equals to $g^{\alpha\beta}$ or to a random element of G.

B. Security

In this section, the proposed solution is proved IND-CKCTA and IND-KGA (indistinguishable against keyword guessing attack).

- ***Confidentiality***: The notion of confidentiality of the EHR in this paper means that the private documents of users must be kept secret from both unauthorized system visitors and the EHR cloud service provider. By this scheme, the health information is protected by means of a strong encryption primitive. The indexes of the conjunctive keywords are encrypted by the dPECK or Re-dtPECK algorithms before uploaded to the cloud server. The service provider could not recover the plaintext of the encrypted data. The keyword extraction from EHR is controlled by the patient and encrypted locally with patient R_i's own secret key. The ciphertext can be derived with R_i's private key sk_{R_i}. However, the server could not get any information about the patients' private keys for generating trapdoors and decrypting the protected documents.

Theorem 1: The proposed Re-dtPECK scheme is IND-CKCTA in the standard model assuming that the truncated decisional l-ABDHE problem and the DBDH problem are intractable.

Proof: The proof of Theorem 1 is in Appendix. B.

Theorem 2: The proposed Re-dtPECK scheme is IND-KGA secure in the standard model assuming that the DDH problem are intractable.

Proof: The proof of Theorem 2 is in Appendix. B.

Hint: There are two types of attackers in IND-CKCTA security proof including the server and the outside attacker. IND-CKCTA ensures that the server could not distinguish the ciphertext encrypts from keyword set and time if the trapdoor for given keywords and time is not obtained. On the other hand, the outside attacker could not make decisions about the ciphertext of certain keywords and time without the server's private key even though all the trapdoors for the other keywords and times are available.

IND-KGA guarantees that the attackers including the server attackers and outside attackers could not find the relationship between the given trapdoor and the challenge keywords even though other trapdoors for both delegator and delegatee can be obtained.

Since the first PEKS scheme proposed by Boneh *et al.* [4], almost all searchable encryption schemes in public key setting

TABLE II
FEATURES COMPARISON WITH OTHER SCHEMES

Scheme	F1	F2	F3	F4	F5	F6	F7	F8
[4]	No	No	No	No	No	No	Yes	Yes
[6]	No	No	Yes	No	--	No	Yes	Yes
[8]	No	Yes	Yes	No	Yes	No	Yes	Yes
[9]	No	Yes	No	No	No	No	Yes	Yes
[10]	No	Yes	No	No	No	No	Yes	Yes
[13]	No	No	No	Yes	No	No	Yes	Yes
[14]	No	No	No	Yes	No	No	Yes	Yes
[15]	No	No	No	Yes	No	No	Yes	Yes
[16]	No	Yes	No	Yes	No	No	Yes	Yes
[21]	No	No	Yes	No	No	No	Yes	Yes
[22]	No	No	Yes	No	Yes	No	Yes	Yes
[23]	No	No	Yes	No	No	No	Yes	Yes
[24]	No	No	Yes	No	--	No	Yes	Yes
[25]	No	No	Yes	No	Yes	No	Yes	Yes
[7]	No	No	Yes	No	Yes	No	Yes	Yes
[28]	Yes	No	No	No	Yes	No	Yes	Yes
[29]	Yes	No	No	No	--	No	Yes	Yes
[30]	Yes	No	No	No	No	No	Yes	Yes
[33]	No	Yes	--	No	--	Yes	No	No
[34]	No	Yes	--	No	--	Yes	No	Yes
Ours	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**	**No**	**Yes**	**Yes**

Note:
F1: Time control F2: Conjunctive keywords F3: Against KG attack
F4: Proxy search F5: Standard model F6: Boolean query
F7: No key sharing F8: Dynamic data change

do not include test queries in security proof. The reason is that the test algorithm can be run if the keyword trapdoor and ciphertext are obtained. In fact, the adversary is able to get the trapdoors through trapdoor queries for the delegator or delegatee. The target ciphertext can be obtained in challenge phase. In PEKS schemes without designated tester, the test algorithm can be run by any attacker. In this work, the test algorithm can only be executed by the data server using his private key, which is the solid meaning of "designated tester".

VI. PERFORMANCE ANALYSIS

Security level, efficiency and the utility function are important indicators to evaluate whether a scheme is suitable for the privacy–preserving in the EHR cloud storage. The proposed Re-dtPECK will be compared with other relevant schemes according to these indicators. A simulation result on an experimental test-bed is also provided to measure the performance of Re-dtPECK scheme.

A. Comparison

In this subsection, we have compared our proposed Re-dtPECK scheme with other searchable encryption schemes in [4], [6]–[10], [13]–[16], [21]–[25], [33], and [34] and the relevant proxy re-encryption schemes in [28]–[30] in terms of functionality, communication and computation overhead. The comparison results are shown in Table II and Table III.

The comparison in Table II indicates that the proposed scheme has versatile functions and higher security level.

- ***Time Control***: In privacy-preserving keyword query systems, the time controlled function will enable the EHR data owner to flexibly disseminate his access right to another user in a specified period of time. The delegation revocation can be achieved even when the EHR data owner is offline. Only some proxy re-encryption schemes in [28]–[30] without a query function have addressed it. Unfortunately, all the available searchable encryption schemes cannot hold this feature.
- ***Conjunctive Keywords***: Compared with the single keyword search, the conjunctive keyword search function provides the users more convenience to return the accurate results that fulfills users' multiple requirements. The users do not have to query an individual keyword and rely on an intersection calculation to obtain what they needs. To the best of our knowledge, there is no existing proxy re-encryption searchable encryption scheme could provide the conjunctive keywords search capability without requiring a random oracle. Our scheme has solved this open problem. The scheme in [16] could provide both the conjunctive keywords search and the delegation function. Unfortunately, it is proved in the random oracle (R.O.) model, which greatly impairs the security level.
- ***Against off-line KG attack***: Up to date, there is no effective approach to prevent the eavesdropping attacks over a public communication channel. An outside attacker could easily implement an off-line KG attack by monitoring the information channel between the patient and the data center if no powerful countermeasure is applied. Most of the existing schemes have not taken KG attacks into consideration. The schemes in [4], [9], [10], [13]–[16], and [28]–[30] could not resist KG attacks. The proposed scheme is immune to KG attacks, which is proved in Theorem 2.
- ***Proxy***: The proxy re-encryption technology is practical in EHR systems. It will greatly facilitate patient delegating the search and access rights. Schemes in [4], [6]–[10], [21]–[25], [28]–[30], [33] and [34] could not provide the proxy re-encryption searchable encryption function to the users.
- ***Standard model*****:** There is no formal security proof on the schemes in [6], [24], and [29]. The schemes in [4], [9], [10], [13]–[16], [21], [23], and [30] have been proved to be secure relying on R.O. model. However, it could not demonstrate the security functionality in a real life. It is shown in [17] and [18] that the schemes might lead to insecure schemes when the random oracles are substituted with concrete hash functions. Thus, it is better to design a secure primitive that does not rely on the random oracle model. Our scheme is proved secure based on the standard model, which has a higher level security.
- ***No key sharing in multi-client setting***: The schemes in [33] and [34] are constructed in symmetric searchable encryption (SSE) setting, which allows a single user to implement upload or query operations. In order to be used in multi-client setting, the scheme has to make the symmetric key be shared by authorized users. Other schemes built in PEKS setting do not suffer such problem.
- ***Dynamic data change***: The scheme in [34] can support additions as well as deletions on encrypted data without

TABLE III
COMMUNICATION AND COMPUTATION OVERHEAD COMPARISON WITH OTHER SCHEMES

Scheme	Communication Overhead							Computation Overhead						
	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	T12	T13	T14
[8]	**Yes**	2	1	l +2	l+2	--	--	t_{Pair}+2t_{Exp}	$(l+1)t_{Exp}$	--	2t_{Exp}	--	3t_{Pair}	--
[9]	**Yes**	3l +3	3l +2	2l +2	2l +1	--	--	t_{Pair}+(3l+2)t_{Exp}	$(3l+2)t_{Exp}$	--	(5l+1)t_{Exp}	--	(2l+1)t_{Pair}	--
[10]	**Yes**	l +5	1	2l +4	l +3	--	--	t_{Exp}	t_{Pair}+(2l+4)t_{Exp}	--	(3l+5)t_{Exp}	--	(2l+1)t_{Pair}	--
[12]	**Yes**	1	1	l+1	l+1	--	--	2t_{Exp}	$(2l+1)t_{Exp}$	--	t_{Exp}	--	2t_{Pair}	--
[13]	No	1	1	7	1	7	--	t_{Exp}	2t_{Pair}+5t_{Exp}	t_{Exp}	t_{Exp}	--	t_{Pair}	--
[14]	No	1	1	2	2	2	--	t_{Exp}	t_{Pair} + 3t_{Exp}	t_{Exp}	3t_{Exp}	--	t_{Pair}	--
[15]	No	5	5	7	4	4	--	5t_{Exp}	3t_{Pair}+3t_{Exp}	t_{Pair}+2t_{Exp}	2t_{Exp}	--	t_{Pair}	--
[16]	**Yes**	l +3	3	2l +4	3	l +3	--	$2(l+3)t_{Exp}$	$(4l+3)t_{Exp}$	2t_{Pair}	t_{Exp}	--	2t_{Pair}	--
[22]	No	6	1	4	2	--	--	t_{Exp}	t_{Pair} + 6t_{Exp}	--	4t_{Exp}	--	2t_{Pair}	--
[23]	No	1	1	2	2	--	--	2t_{Exp}	t_{Pair} + 2t_{Exp}	--	3t_{Exp}	--	t_{Pair}	--
[24]	No	3	1	2	2	--	--	4t_{Exp}	3t_{Pair}+2t_{Exp}	--	3t_{Exp}	--	t_{Pair}	--
[25]	No	1	1	2	2	--	--	4t_{Exp}	4t_{Pair}+5t_{Exp}	--	3t_{Exp}	--	t_{Pair}	--
[7]	No	l +3	l +3	7	2	--	--	$(l+3)t_{Exp}$	3t_{Pair}+8t_{Exp}	--	t_{Exp}	--	3t_{Pair}	--
[28]	No	1	1	7	--	9	--	t_{Exp}	4t_{Pair} + 8t_{Exp}	4t_{Exp}	--	--	--	--
[29]	No	3	2	N+2	--	3N+2	--	5t_{Exp}	t_{Pair}+(N+1)t_{Exp}	t_{Pair}+(6N+1)t_{Exp}	--	--	--	--
[30]	No	1	1	7	--	7	--	t_{Exp}	t_{Pair} + 7t_{Exp}	7t_{Exp}	--	--	--	--
Ours	**Yes**	**1**	**1**	$\boldsymbol{l}$ **+3**	$\boldsymbol{l}$ **+3**	$\boldsymbol{l}$ **+6**	$\boldsymbol{l}$ **+4**	$\boldsymbol{t_{Exp}}$	$(\boldsymbol{l}+4)\boldsymbol{t_{Exp}}$	$(\boldsymbol{l}+5)\boldsymbol{t_{Exp}}$	$(\boldsymbol{l}+1)\boldsymbol{t_{Exp}}$	$(\boldsymbol{l}+2)\boldsymbol{t_{Exp}}$	$(\boldsymbol{l}+2)\boldsymbol{t_{Pair}}$	$(\boldsymbol{l}+4)\boldsymbol{t_{Pair}}$

Note: T1: Conjunctive keywords; T2:Public key size; T3: Private key size (user); T4: Ciphertext size (delegator); T5: Tradoor size (delegator); T6: Re-encryption ciphertext size; T7: Trapdoor size (delegatee); T8: Key Generation; T9: Encryption; T10: Re-encryption; T11: Trapdoor (delegator); T12: Trapdoor (delegatee); T13: Test (delegator); T14: Test (delegatee)
l: size of keyword set N: number of conjunctive clauses in the access structure [29]
t_{Exp} : execution time for exponentiation operation; t_{Pair} : execution time for bilinear pairing operations.

using the same symmetric key, while the scheme in [33] cannot. The other schemes that are built in PEKS setting can also implement additions and deletions operations through using data owner's public and private keys.

Thus, the proposed scheme has various useful functions and has stronger security functionality than those of most of the existing searchable encryption schemes.

The communication overhead comparison in Table III shows that the proposed Re-dtPECK scheme has a small size of public key and private key.

- The sizes of the public keys in the schemes in [7], [9], [10], and [16] are linear with the number l of the conjunctive keywords, while the size of the public keys in this Re-dtPECK scheme is only one element in group G.
- The size of the private keys in the scheme in [7] and [9] is linear with the number l of the conjunctive keywords, while the size of the private keys in our scheme is only one element in Z_p^*.
- The size of the ciphertext in the schemes in [8]–[10], [12], and [16] are $l+2$, $2l+2$, $2l+4$, $l+1$, $2l+4$, respectively, while the size of the ciphertext is $l+3$ in our scheme.
- The sizes of the trapdoor in the schemes in [8]–[10] and [12] and ours are linear with the number l of the conjunctive keywords.

Note: It seems that the schemes in [13]–[15], [22]–[25], [28], and [30] have a better efficiency of the ciphertext than our scheme. However, these schemes can only support the single keyword search, while our scheme is more efficient superior than other searchable encryption schemes in the sense of that it allows the conjunctive keyword search.

The computation overhead comparison is also shown in Table III. There are two major time-consuming computation operations: exponentiation and bilinear pairing. The bilinear pairing computation is the most time consuming operation among all computations. It can be found that the proposed Re-dtPECK can achieve a higher efficiency than other PEKS schemes that supports conjunctive keyword search.

B. Efficiency Evaluation by Simulation

We have evaluated the proposed Re-dtPECK scheme by implementing key components on an experimental workbench, including the system global setup, the key generation, the re-encryption key generation, the trapdoor generation and the test algorithms. The pairing-based cryptography (PBC) Library [32] is used. We have selected the type-A elliptic curve parameter, which provides 1024-bit discrete log security strength equivalent to the group order of 160-bit. The experiments have been executed on a PC running Windows7 with an Intel core CPU at 2.5GHz and a 4.0 GB of the memory. We have evaluated the communication, the computation and

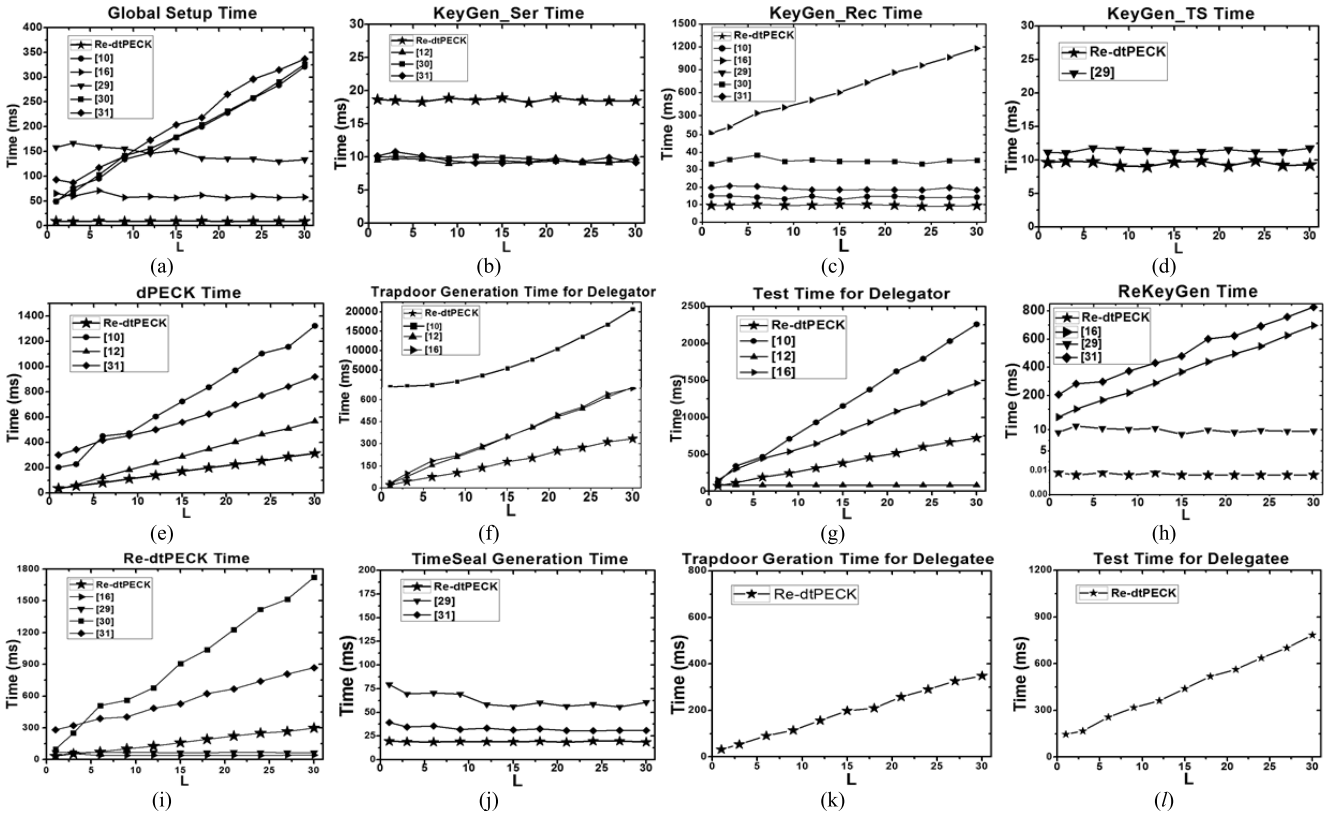

Fig. 4. Time Efficiency of the Scheme. (a) Global Setup Time. (b) Key Generation Time for Server. (c) Key Generation Time for Receiver. (d) Key Generation Time for TS. (e) dPECK Time. (f) Trapdoor Generation Time for Delegator. (g) Test Time for Delegator. (h) Re-encryption Key Generation Time. (i) Re-dtPECK Time. (j) TimeSeal Generation Time. (k) Trapdoor Generation Time for Delegatee. (*l*) Test Time for Delegatee.

the storage overhead of the proposed scheme. To the best of our knowledge, there is no existing work with the searchable encryption and delegation function to provide such experimental results.

Fig. 4 shows the operation time of each algorithm of our scheme with the different values of l, which is the number of keywords. We have also tested the schemes in [10], [12], [16], and [28]–[30] on the same experimental workbench so that a simulation comparison can be made fair. In order to describe the performance differences, a non-uniform axis has been used in Fig. 4 (c, f, h) to make the distinct values much clearer. Since the schemes in [10], [12], [16], and [28]–[30] do not have $Trapdoor_R$ and $Test_R$ algorithms, there is only one line in Fig. 4 (k, l). Compared with these schemes, it is easy to find that our scheme almost always has a smaller execution time. In Fig.4(b), our scheme takes 9ms more than other schemes due to that one more group element is randomly selected for generating the public key of the data server. This tiny difference can be virtually ignored because key generation algorithm is executed only once by the TTP.

In our *dPECK* algorithm, the ciphertext is computed as $t = e(X, V)^s, C_1 = t \cdot e(g, g)^{-r}, C_2 = g^s, B_\psi = g^{r \cdot \eta_\psi}, 0 \leq \psi \leq l$. Two bilinear pairing operations $e(g, g)$ and $e(X, V)$ can be pre-computed, while they will not consume additional computation time in the document encryption phase. The execution times of the *GlobalSetup*, $KeyGen_{Ser}$, KeyGen_{Rec}, and $KeyGen_{TS}$ algorithms are approximately 9ms, 18ms, 10ms and 9ms. The execution times of the *ReKeyGen* and *TimeSeal* algorithms are about 0.01ms and 19ms. The execution time of *Trapdoor*, $Trapdoor_R$, *Test*, $Test_R$, dPECK and Re-dtPECK algorithm grows with the number l of keywords that is extracted from patient's EHR document.

The experimental results indicate that the proposed scheme can achieve a high efficiency, which is desirable for the real world EHR cloud applications.

VII. Conclusion

In this paper, we have proposed a novel Re-dtPECK scheme to realize the timing enabled privacy-preserving keyword search mechanism for the EHR cloud storage, which could support the automatic delegation revocation. The experimental results and security analysis indicate that our scheme holds much higher security than the existing solutions with a reasonable overhead for cloud applications. To the best of our knowledge, until now this is the first searchable encryption scheme with the timing enabled proxy re-encryption function and the designated tester for the privacy–preserving EHR cloud record storage. The solution could ensure the confidentiality of the EHR and the resistance to the KG attacks. It has also been formally proved secure based on the standard model under the hardness assumption of the truncated decisional l-ABDHE problem and the DBDH problem. Compared with other classical searchable encryption schemes, the efficiency analysis shows that our proposed scheme can achieve high computation and storage efficiency besides its higher security.

Our simulation results have also shown that the communication and computation overhead of the proposed solution is feasible for any real world application scenarios.

APPENDIX

A. Security Model of Re-dtPECK

The security model of indistinguishable against keyword guessing attack (IND-KGA) is in accordance with [7] and [22], which is omitted here due to the length limitation. The only difference is that the trapdoor for delegatee is also considered in our proof (Please refer to Appendix. B for the detailed proof).

A Re-dtPECK scheme is IND-CKCTA secure if no probabilistic polynomial time (PPT) attacker $\mathcal{A}$ can win the games below with non-negligible advantage. In the game, $\mathcal{C}$ is the game challenger, k is the security parameter. We consider the following two games.

Game 1: $\mathcal{A}$ is assumed to be a server.

- ***Setup***: The *GlobalSetup*, $KeyGen_{Ser}$, $KeyGen_{Rec}$ and $KeyGen_{TS}$ algorithms are executed. The global parameter GP and keys (sk_S, pk_S), $(sk_R, pk_R)(sk_{TS}, pk_{TS})$ are generated. $(GP, sk_S, pk_S, pk_R, pk_{TS})$ are given to $\mathcal{A}$ while sk_R is kept secret from $\mathcal{A}$.
- ***Phase 1***: $\mathcal{A}$ makes the following queries.

  (1) Time seal query: $\mathcal{A}$ adaptively queries a time T. $\mathcal{C}$ runs *TimeSeal* and generates the time seal S_T, which is given to $\mathcal{A}$.

  (2) Trapdoor queries: $\mathcal{A}$ adaptively queries a number of keyword sets (denoted as Q) to generate a trapdoor for the user R_i. $\mathcal{C}$ runs $Trapdoor$ and $T_{Q,I}$ is generated and given to $\mathcal{A}$.

  If delegation indicator θ equals to 1, run the queries (3) to (5).

  (3) Trapdoor queries for delegatee: $\mathcal{A}$ adaptively queries several keyword sets (denoted as Q) to generate a trapdoor for delegatee R_j. $\mathcal{C}$ runs $Trapdoor_R$ and $T_{Q,J}$ is generated and given to $\mathcal{A}$.

  (4) Re-encryption key queries: $\mathcal{A}$ adaptively queries the receivers' ID tuple (i, j), where $i \neq j$. Challenger $\mathcal{C}$ runs *ReKeyGen* and returns a re-encryption key $rk_{R_i \to R_j}$ to the adversary.

  (5) Re-encryption queries: $\mathcal{A}$ adaptively queries (i, j) and the original dPECK ciphertext C_I. Challenger $\mathcal{C}$ returns the re-encrypted ciphertext $C_J \leftarrow Re - dtPECK$ to the adversary.
- ***Challenge***: $\mathcal{A}$ outputs a challenge set of keywords (W_0^*, W_1^*) and the time (T_0^*, T_1^*). Upon receiving them, the challenger chooses $\sigma, \upsilon \in \{0, 1\}$ uniformly at random and creates the target ciphertext C^* and returns it to $\mathcal{A}$.
- ***Phase 2***: Adversary $\mathcal{A}$ is allowed to ask the same types of queries as in phase 1, except the following queries.

  (1) Time seal queries: $TimeSeal(GP, sk_{TS}, T_\upsilon^*, pk_{R_i}, pk_{R_j})$ is not allowed to be queried, where $\upsilon \in \{0, 1\}$.

  (2) Trapdoor queries: $Trapdoor(GP, pk_S, sk_{R_i}, Q)$ is not allowed to be queried if the generated trapdoor is distinguishable for W_0^* and W_1^*.

  (3) Trapdoor queries for delegatee: $Trapdoor_R$ $(GP, pk_S, sk_{R_j}, Q, S_{T_\upsilon^*})$ is not allowed to be queried if the generated trapdoor is distinguishable for W_0^* and W_1^*.
- ***Guess***: $\mathcal{A}$ outputs its guess σ', $\upsilon' \in \{0, 1\}$ and wins the game if $\sigma' = \sigma$, $\upsilon' = \upsilon$.

Game 2: $\mathcal{A}$ is assumed to be an outside attacker including the receiver.

- ***Setup***: The *GlobalSetup*, $KeyGen_{Ser}$, $KeyGen_{Rec}$ and $KeyGen_{TS}$ algorithms are executed. The global parameter GP and keys (sk_S, pk_S), $(sk_R, pk_R)(sk_{TS}, pk_{TS})$ are generated. $(GP, pk_S, pk_R, sk_R, pk_{TS})$ are given to $\mathcal{A}$ while sk_S is kept secret from $\mathcal{A}$.
- ***Phase 1***: $\mathcal{A}$ makes the following queries.

  (1) Time seal query: $\mathcal{A}$ adaptively queries a time T, challenger $\mathcal{C}$ runs *TimeSeal* and generates the time seal S_T, which is given to $\mathcal{A}$.

  If delegation indicator θ equals to 1, run the queries (2) to (3).

  (2) Re-encryption key queries: $\mathcal{A}$ adaptively queries the receivers' ID tuple (i, j), where $i \neq j$. Challenger $\mathcal{C}$ runs *ReKeyGen* and returns the re-encryption key $rk_{R_i \to R_j}$ to the adversary.

  (3) Re-encryption queries: $\mathcal{A}$ adaptively queries (i, j) and the original dPECK ciphertext C_I. Challenger $\mathcal{C}$ returns the re-encrypted ciphertext $C_J \leftarrow Re - dtPECK$ to the adversary.
- ***Challenge***: $\mathcal{A}$ outputs the target keyword pair (W_0^*, W_1^*) and the time pair (T_0^*, T_1^*). Upon receiving them, the algorithm chooses $\sigma, \upsilon \in \{0, 1\}$ uniformly at random and creates the target ciphertext C^* and returns it to $\mathcal{A}$.
- ***Phase 2***: Adversary $\mathcal{A}$ is allowed to ask the same types of queries as in phase 1, except the following queries.

  (1) Time seal queries: $TimeSeal(GP, sk_{TS}, T_\upsilon^*, pk_{R_i}, pk_{R_j})$ is not allowed to be queried, where $\upsilon \in \{0, 1\}$.
- ***Guess***: $\mathcal{A}$ outputs its guess σ', $\upsilon' \in \{0, 1\}$ and wins the game if $\sigma' = \sigma$, $\upsilon' = \upsilon$.

B. Security Proof of Re-dtPECK

Theorem 1: Suppose the truncated decisional l-ABDHE and DBDH assumption holds, the proposed scheme is IND-CKCTA secure in the standard model. If there is a PPT adversary $\mathcal{A}$, who breaks (t, ε)- IND-CKCTA security of the proposed Re-dtPECK scheme, then we can construct a PPT adversary $\mathcal{C}$ to solve the (t_1, ε_1) truncated decisional l-ABDHE and (t_2, ε_2) DBDH problem with

$$\varepsilon_1 \geq \varepsilon, t_1 \geq t + t_e[(2l+3)q_{Trapdoor} + (l+5)q_{Re-dtPECK} + 2q_{TimeSeal} + (2l+4)q_{Trapdoor_R}],$$

$$\varepsilon_2 \geq \varepsilon, t_2 \geq t + t_e[2q_{TimeSeal} + (l+5)q_{Re-dtPECK}],$$

where t_e denotes the running time of an exponentiation, $q_{Trapdoor}$, $q_{Re-dtPECK}$, $q_{TimeSeal}$, $q_{Trapdoor_R}$ denotes the number of the $Trapdoor$, $Re-dtPECK$, $TimeSeal$, $Trapdoor_R$ queries.

Game 1: $\mathcal{A}$ is assumed to be a server.

Proof: Let k be the security parameter. Suppose that $\mathcal{A}$ is given a tuple $Tu = (g', g'_{l+2}, g, g_1, \cdots, g_l, Z)$ as an instance for the truncated l-ABDHE problem, where Z is either $e(g_{l+1}, g')$ or a random element inG_1 and $g_i = g^{\alpha^i}, g'_i = (g')^{\alpha^i}$. The game between $\mathcal{A}$ and $\mathcal{C}$ proceeds as follows.

- ***Setup***: Challenger $\mathcal{C}$ chooses $\beta \in Z_p^*, V \in G_1$ at random and computes $X = g^\beta$. Let $pk_S = (V, X), sk_S = (\beta)$ be the data server's public and private key, respectively. Define $Y = g^\alpha$. The receiver's public and private key pair will be $pk_R = Y, sk_R = \alpha$. $\mathcal{C}$ chooses τ at random as the private key sk_{TS} of the TS and computes $pk_{TS} = g^\tau$. Send $(pk_S, sk_S, pk_R, pk_{TS})$ to $\mathcal{A}$ while sk_R is kept secret from $\mathcal{A}$.
- ***Phase 1***: $\mathcal{A}$ makes the following queries.

  (1) Time seal query: $\mathcal{A}$ issues time seal queries for time T. $\mathcal{C}$ randomly chooses $r_T \in \mathbb{Z}_p^*$, computes $h_T = (Y \cdot g^{-r_T})^{1/[\tau - H(T, pk_i)]}$ and sends $S_T = (r_T, h_T)$ to $\mathcal{A}$.

  (2) Trapdoor queries: $\mathcal{A}$ adaptively queries a keyword set $Q_i = (w_{i,1}, \cdots, w_{i,m})$, $m \le l$, to obtain a trapdoor for the user R_i. $\mathcal{C}$ randomly chooses $T_{Q_i,-1}, \varsigma \in \mathbb{Z}_p^*$, computes $T_{Q,-2} = g^\varsigma, T_{Q,\psi} = g^{m^{-1} \cdot T_{Q,-1} \cdot (y_i)^\psi \cdot \sum_{\mu=1}^m H(w_{\gamma\mu})^\psi} \cdot X^\varsigma$, $0 \le \psi \le l$. Since

$$T_{Q,\psi} = g_\psi^{m^{-1} \cdot T_{Q_i,-1} \cdot \sum_{\mu=1}^m H(w_{i,\mu})^\psi}$$
$$X^\varsigma = g^{m^{-1} \cdot T_{Q,-1} \cdot \alpha^\psi \cdot \sum_{\mu=1}^m H(w_{i,\mu})^\psi} X^\varsigma$$

  $\mathcal{C}$ has successfully simulated the trapdoor.

  If delegation indicator θ equals 1, run the queries (3) to (5).

  (3) Trapdoor queries for delegatee: $\mathcal{A}$ adaptively queries a keyword set $Q_i = (w_{i,1}, \cdots, w_{i,m})$, $m \le l$. $\mathcal{C}$ randomly chooses $r_T, \varsigma \in \mathbb{Z}_p^*$ and sets:

$$T_{Q,-1} = r_T, \ T_{Q,-2} = g^\varsigma,$$
$$T_{Q_i,-3} = (g_2 \cdot Y^{-r_T})^{1/[\tau - H(T, pk_i)]},$$
$$T_{Q,\psi} = g_\psi^{m^{-1} \cdot r_T \cdot \sum_{\mu=1}^m H(w_{i,\mu})^\psi} X^\varsigma \ (0 \le \psi \le l).$$

  Since $(g_2 \cdot Y^{-r_T})^{\frac{1}{\tau - H(T, pk_i)}} = (Y \cdot g^{-r_T})^{\frac{\alpha}{\tau - H(T, pk_i)}}$, this is a valid trapdoor for Q_i. Thus, $\mathcal{C}$ has successfully simulated the trapdoor $T_{Q_i,J}$ for the delegatee.

  (4) Re-encryption key queries: $\mathcal{A}$ adaptively queries receivers' ID tuple (i, j), where $i \ne j$. $\mathcal{C}$ runs $KeyGen_{Rec}$ to obtain y_i and y_j. Then, the challenger runs *ReKeyGen* and returns a re-encryption key $rk_{R_i \to R_j} = y_i / y_j$ to the adversary.

  (5) Re-encryption queries: $\mathcal{A}$ adaptively queries (i, j) and the original dPECK ciphertex C_I. $\mathcal{C}$ runs *ReKeyGen* and obtains the re-encryption key $rk_{R_i \to R_j} = y_i / y_j$. Then, the challenger runs the re-encryption algorithm, computes $C_3 = (g^{-H(T, pk_i)} pk_{TS})^\mu, C_4 = Y_j^\mu$, $C_5 = \mathrm{g}^\mu, B'_\psi = B_\psi^{(rk_{R_i \to R_j})^\psi}$ $(0 \le \psi \le l)$ and returns C_J to $\mathcal{A}$.
- ***Challenge***: $\mathcal{A}$ outputs a challenge set of keywords (W_0^*, W_1^*) and the time (T_0^*, T_1^*), $\mathcal{C}$ randomly chooses $\sigma, \upsilon \in \{0, 1\}$, $W_\sigma^* = (w_{\sigma,1}^*, \cdots, w_{\sigma,l}^*)$, and T_υ^* as the target. Let $f(x) = x^{l+2}$ and $F(x) = (f(x) - f(\prod_{\psi=1}^m H(w_{\sigma,\psi}^*)))/x$, which is a polynomial of degree $l + 1$. $\mathcal{C}$ randomly selects $s^*, \mu, x_{\sigma,j} \in Z_p^*$ for$1 \le j \le l$ and computes $t^* = e(X, V)^{s^*}$. $\mathcal{C}$ constructs a polynomial $\Upsilon^*(x) = \eta_l^* x^l + \cdots + \eta_1^* x + \eta_0^*$ such that $x_{\sigma,1}, \cdots, x_{\sigma,l}$ are l roots of the equation $\Upsilon^*(x) = 1$. $\mathcal{C}$ sets

$$C_1^* = t^* \cdot Z^{-1} \cdot e(g', \prod_{\psi=0}^l g^{F_\psi \cdot \alpha^i})^{-1}, \ C_2^* = g^{s^*},$$
$$C_3^* = (g^{-H(T_\upsilon^*, pk_i)} pk_{TS})^\mu, \ C_4^* = Y_j^\mu, \ C_5^* = \mathrm{g}^\mu,$$
$$B_\psi^* = (g')^{F(\alpha) \cdot \eta_\psi^*}, \quad 0 \le \psi \le l.$$

  where F_i is the coefficient of x^i in $F(x)$. Let $r^* = (\log_g g') F(\alpha)$. Since$\log_g g'$ is random, r^* is uniformly random. If $Z = e(g_{l+1}, g')$, then

$$C_1^* = t^* \cdot e(g_{l+1}, g')^{-1} \cdot e(g', \prod_{\psi=0}^l g^{F_\psi \cdot \alpha^i})^{-1}$$
$$= t^* \cdot e(g', \prod_{\psi=0}^{l+1} g^{F_\psi \cdot \alpha^i})^{-1} = t^* \cdot e(g, g)^{-r^*},$$
$$B_\psi^* = (g')^{F(\alpha) \cdot \eta_\psi^*} = g^{r^* \cdot \eta_\psi^*}, \quad 0 \le \psi \le l.$$

  Then, it is a valid encryption for W_σ^* and T_υ^*.
- ***Phase 2***: Adversary $\mathcal{A}$ is allowed to ask the same types of queries as in phase 1, except the following queries.

  (1) Time seal queries: $TimeSeal(GP, sk_{TS}, T_\upsilon^*, pk_{R_j})$ is not allowed to be queried, where $\upsilon \in \{0, 1\}$.

  (2) Trapdoor queries: $Trapdoor(GP, pk_S, sk_{R_i}, Q)$ is not allowed to be queried if the generated trapdoor is distinguishable for W_0^* and W_1^*.

  (3) Trapdoor queries for delegatee: $Trapdoor_R$ $(GP, pk_S, sk_{R_j}, Q, S_{T_\upsilon^*})$ is not allowed to be queried if the generated trapdoor is distinguishable for W_0^* and W_1^*.
- ***Guess***: $\mathcal{A}$ outputs a guess σ', $\upsilon' \in \{0, 1\}$. If $\sigma' = \sigma$, $\upsilon' = \upsilon$, then $\mathcal{C}$ outputs 1 meaning $Z = e(g_{l+1}, g')$. Otherwise, it outputs 0 meaning $Z \ne e(g_{l+1}, g')$ but a random element in G_1.

Probability Analysis: In the guess phase, if the adversary is capable to break the scheme with the advantage ε, $e(g_{l+1}, g')$ will appear in the tuple list with probability $1/2 + \varepsilon$ at least. Then, the advantage of $\mathcal{C}$ against the truncated decisional l-ABDHE problem is at least $\varepsilon_1 \ge \varepsilon$.

Time Analysis: The execution time of the simulation is dominated by the exponent operation in the query phase. Then

$$t_1 \ge t + t_e[(2l + 3)q_{Trapdoor} + (l + 5)q_{Re-dtPECK} + 2q_{TimeSeal} + (2l + 4)q_{Trapdoor_R}]$$

Game 2: $\mathcal{A}$ is assumed to be an outside attacker including the receiver.

Proof: Suppose that the attacker $\mathcal{A}$ is given a tuple $Tu = (g, g^a, g^b, g^c, Z)$ as an instance for the DBDH problem, where Z is either $e(g, g)^{abc}$ or a random element of G_1. The game between $\mathcal{A}$ and $\mathcal{C}$ proceeds as follows.

- ***Setup***: Let $X = g^a, V = g^b$, the data server's public and private key pair is $pk_S = (V, X)$ and $sk_S = a$. $\mathcal{C}$ randomly chooses $y \in Z_p^*$ and calculates $Y = g^y$.

The receiver's public and private key pair is $pk_R = Y$, $sk_R = y$. $\mathcal{C}$ chooses τ at random as sk_{TS} and computes $pk_{TS} = g^{\tau}$. Then, $(pk_S, pk_R, sk_R, pk_{TS})$will be sent to $\mathcal{A}$ while sk_S is kept secret from $\mathcal{A}$.

- ***Phase 1***: $\mathcal{A}$ makes the following queries.

  (1) Time seal query: $\mathcal{A}$ issues time seal queries for time T. $\mathcal{C}$ randomly chooses $r_T \in Z_p^*$, computes $h_T = (Y \cdot g^{-r_T})^{1/[\tau - H(T, pk_i)]}$ and sends $S_T = (r_T, h_T)$ to $\mathcal{A}$.

  If delegation indicator θ equals 1, run the queries (2) to (3).

  (2) Re-encryption key queries: $\mathcal{A}$ adaptively queries receivers' ID tuple (i, j), $i \neq j$. $\mathcal{C}$ runs *ReKeyGen* and returns $rk_{R_i \to R_j} = y_i / y_j$ to $\mathcal{A}$.

  (3) Re-encryption queries: $\mathcal{A}$ adaptively queries (i, j) and the original dPECK ciphertex C_I. $\mathcal{C}$ runs *ReKeyGen* and obtains $rk_{R_i \to R_j} = y_i / y_j$. $\mathcal{C}$ computes $C_3 = (g^{-H(T, pk_i)} pk_{TS})^{\mu}, C_4 = Y_j^{\mu}, C_5 = \mathrm{g}^{\mu}$, $B'_{\psi} = B_{\psi}^{(rk_{R_i \to R_j})^{\psi}} (0 \leq \psi \leq l)$ and returns the re-encrypted ciphertext C_J to $\mathcal{A}$.
- ***Challenge***: $\mathcal{A}$ outputs a challenge set of keywords (W_0^*, W_1^*) and time pair (T_0^*, T_1^*), $\mathcal{C}$ randomly chooses $\sigma, \upsilon \in \{0, 1\}$ and $W_{\sigma}^* = (w_{\sigma,1}^*, \cdots, w_{\sigma,l}^*)$, T_{υ}^* as the target. $\mathcal{C}$ constructs a polynomial $\Upsilon(x) = \eta_l^* x^l + \eta_{l-1}^* x^{l-1} + \cdots + \eta_1^* x + \eta_0^*$ such that $y_i H(w_{\sigma,1}^*), \cdots, y_i H(w_{\sigma,l}^*)$ are l roots of the equation $\Upsilon(x) = 1$. $\mathcal{C}$ randomly selects $r^*, \mu \in Z_p^*$ and sets

$$t^* = Z, C_1^* = Z \cdot e(g, g)^{-r^*}, C_2^* = g^c, B_{\psi}^* = g^{r^* \cdot \eta_{\psi}^*},$$
$$C_3^* = (g^{-H(T_{\upsilon}^*, pk_i)} pk_{TS})^{\mu}, C_4^* = Y_j^{\mu}, C_5^* = \mathrm{g}^{\mu}.$$

  If $Z = e(g, g)^{abc}$, then $t^* = e(g, g)^{abc} = e(X, V)^c$. It is a valid encryption for W_{σ}^* and T_{υ}^*.
- ***Phase 2***: Adversary $\mathcal{A}$ is allowed to ask the same types of queries as in phase 1, except the following queries.

  (1) Time seal queries: $TimeSeal(GP, sk_{TS}, T_{\upsilon}^*, pk_{R_i}, pk_{R_j})$ is not allowed to be queried, where $\upsilon \in \{0, 1\}$.
- ***Guess***: $\mathcal{A}$ outputs a guess $\sigma', \upsilon' \in \{0, 1\}$. If $\sigma' = \sigma$, $\upsilon' = \upsilon$, then $\mathcal{C}$ outputs 1 meaning $Z = e(g, g)^{abc}$. Otherwise, it outputs 0 meaning $Z \neq e(g, g)^{abc}$ but a random element in G_1.

Probability Analysis: In the guess phase, if the adversary is capable to break the scheme with the advantage ε, $e(g, g)^{abc}$ will appear in the tuple list with probability $1/2 + \varepsilon$ at least. Then, the advantage of $\mathcal{C}$ against the DBDH problem is at least $\varepsilon_2 \geq \varepsilon$.

Time Analysis: The execution time of the simulation is dominated by the exponent operation in the query phase. Then we have $t_2 \geq t + t_e[2q_{TimeSeal} + (l + 5)q_{Re-dtPECK}]$.

This completes the proof of Theorem 1. □

Theorem 2: Suppose the DDH assumption holds, the proposed scheme is IND-KGA secure in the standard model. If there is a PPT adversary $\mathcal{A}$, who breaks (t, ε)-IND-KGA security of the proposed scheme, then we can construct a PPT adversary $\mathcal{C}$ to solve the (t', ε') DDH problem with $\varepsilon' \geq \varepsilon, t' \geq t + t_e[(2l+3)q_{Trapdoor} + (2l+4)q_{Trapdoor_R}]$, where t_e denotes the running time of an exponentiation, $q_{Trapdoor}$, $q_{Trapdoor_R}$ denotes number of $Trapdoor$, $Trapdoor_R$ queries.

Proof: Let k be the security parameter. Suppose that $\mathcal{A}$ is given a tuple $Tu = (g, g^{\alpha}, g^{\beta})$ as an instance for the DDH problem, where Z is either $g^{\alpha\beta}$ or a random element in G. The game between $\mathcal{A}$ and $\mathcal{C}$ proceeds as follows.

- ***Setup***: Challenger $\mathcal{C}$ chooses $V \in G_1$ at random and sets $X = g^{\beta}$. Let $pk_S = (V, X), sk_S = (\beta)$ be the data server's public and private key, respectively. $\mathcal{C}$ randomly chooses $y \in \mathbb{Z}_p^*$ and sets $Y = g^y$. The receiver's public and private key pair will be $pk_R = Y, sk_R = y$. $\mathcal{C}$ chooses τ at random as the private key sk_{TS} of the TS and computes $pk_{TS} = g^{\tau}$. Send (pk_S, pk_R) to $\mathcal{A}$.
- ***Phase 1***: $\mathcal{A}$ makes the following queries.

  (1) Trapdoor queries: $\mathcal{A}$ adaptively queries a keyword set $Q_i = (w_{i,1}, \cdots, w_{i,m})$, $m \leq l$. $\mathcal{C}$ randomly chooses $T_{Q_i,-1}, \varsigma \in \mathbb{Z}_p^*$ and computes $T_{Q,-2} = g^{\varsigma}$, $T_{Q,\psi} = g^{m^{-1} \cdot T_{Q,-1} \cdot y^{\psi} \sum_{\mu=1}^{m} H(w_{\gamma\mu})^{\psi}} \cdot X^{\varsigma}, 0 \leq \psi \leq l$. Then, $\mathcal{C}$ has simulated the trapdoor for the delegator R_i.

  If delegation indicator θ equals 1, run the query (2).

  (2) Trapdoor queries for delegatee: $\mathcal{A}$ adaptively queries a keyword set $Q_i = (w_{i,1}, \cdots, w_{i,m})$, $m \leq l$. $\mathcal{C}$ randomly chooses $r_T, \varsigma \in \mathbb{Z}_p^*$ and sets:

$$T_{Q,-1} = r_T, T_{Q,-2} = g^{\varsigma},$$
$$T_{Q_i,-3} = (g^{y^2} \cdot Y^{-r_T})^{1/[\tau - H(T, pk_i)]},$$
$$T_{Q,\psi} = g^{m^{-1} \cdot r_T \cdot y^{\psi} \sum_{\mu=1}^{m} H(w_{i,\mu})^{\psi}} X^{\varsigma} (0 \leq \psi \leq l).$$

  Since $(g^{y^2} \cdot Y^{-r_T})^{\frac{1}{\tau - H(T, pk_i)}} = (Y \cdot g^{-r_T})^{\frac{y}{\tau - H(T, pk_i)}}$, this is a valid trapdoor for Q_i. Thus, $\mathcal{C}$ has successfully simulated the trapdoor $T_{Q_i,J}$ for the delegatee.
- ***Challenge***: $\mathcal{A}$ outputs a challenge set of keywords (W_0^*, W_1^*). $\mathcal{C}$ randomly selects $\sigma \in \{0, 1\}$, $r_T^* \in Z_p^*$ and sets

$$T_{Q,-1}^* = r_T^*, T_{Q,-2}^* = g^{\alpha},$$
$$T_{Q,\psi}^* = g^{m^{-1} \cdot r_T \cdot y^{\psi} \sum_{\mu=1}^{m} H(w_{\sigma,\mu}^*)^{\psi}} \cdot Z, \quad 0 \leq \psi \leq l.$$

  Then, $T_{Q,I}^* = (T_{Q,-1}^*, T_{Q,-2}^*, T_{Q,\psi}^*)$ is sent to $\mathcal{A}$ as the challenge trapdoor. Let $\varsigma^* = \alpha$. If $Z = g^{\alpha\beta}$, then $T_{Q,-2}^* = g^{\varsigma^*}, T_{Q,\psi}^* = g^{m^{-1} \cdot r_T \cdot y^{\psi} \sum_{\mu=1}^{m} H(w_{\sigma,\mu}^*)^{\psi}} \cdot X^{\varsigma^*}$. Thus, $T_{Q,I}^*$ is a valid trapdoor for W_{σ}^*.
- ***Phase 2***: Adversary $\mathcal{A}$ is allowed to ask the same queries as in phase 1 with the restriction that the trapdoor is not distinguishable for $W_0^*$and W_1^*.
- ***Guess***: $\mathcal{A}$ outputs a guess $\sigma' \in \{0, 1\}$. If $\sigma' = \sigma$, then $\mathcal{C}$ outputs 1 meaning $Z = g^{\alpha\beta}$. Otherwise, it outputs 0 meaning Z is a random element in G.

Probability Analysis: In the guess phase, if the adversary is capable to break the scheme with the advantage ε, $g^{\alpha\beta}$ will appear in the tuple list with probability $1/2 + \varepsilon$ at least. Then, the advantage of $\mathcal{C}$ against the DDH problem is at least $\varepsilon' \geq \varepsilon$.

Time Analysis: The execution time of the simulation is dominated by the exponent operation in the query phase. Then $t' \geq t + t_e[(2l + 3)q_{Trapdoor} + (2l + 4)q_{Trapdoor_R}]$.

This completes the proof of Theorem 2. □

C. Correctness

The test algorithm holds when the *dPECK* ciphertext and the trapdoor match each other and $W \subseteq Q$.

$$
\begin{aligned}
&C_1^{T_{Q,-1}} \prod_{\psi=0}^{l} e[B_\psi, T_{Q,\psi}/(T_{Q,-2})^x] \\
&= \left(te(g,g)^{-r}\right)^{T_{Q,-1}} \prod_{\psi=0}^{l} \\
&\quad \times e\left(g^{r\cdot\eta_\psi}, g^{m^{-1}\cdot T_{Q,-1}\cdot (y_i)^\psi \cdot \sum_{\mu=1}^{m} H(w_{\gamma\mu})^\psi}\right) \\
&= t^{T_{Q,-1}} \cdot e(g,g)^{-r\cdot T_{Q,-1}} \cdot e\,(g,g)^{m^{-1}\cdot r\cdot T_{Q,-1}\cdot \sum_{\mu=1}^{m} \Upsilon[y_i H(w_{\gamma\mu})]} \\
&= t^{T_{Q,-1}}
\end{aligned}
$$

The test algorithm for delegatee still holds when $W \subseteq Q$ and the effective times encapsulated in $T_{Q,J}$ and C_J are accordant.

$$
\begin{aligned}
&C_1^{T_{Q,-1}} e(T_{Q,-3}, C_3) \prod_{\psi=0}^{l} e(B'_\psi, T_{Q,\psi}/(T_{Q,-2})^x) \\
&\quad = t^{T_{Q,-1}} \cdot e(g,g)^{-r\cdot T_{Q,-1}} \cdot e\left((Y_j \cdot g^{-r_T})^{y_j}, g^{\mu}\right) \cdot \\
&e\,(g,g)^{m^{-1}\cdot r\cdot T_{Q,-1}\cdot \sum_{\mu=1}^{m} \sum_{\psi=0}^{l} \eta_\psi \cdot [y_i H(w_{\gamma\mu})]^\psi} \\
&\quad = t^{T_{Q,-1}} \cdot e\left(Y_j, C_4 \cdot C_5^{-T_{Q,1}}\right)
\end{aligned}
$$

REFERENCES

[1] J. C. Leventhal, J. A. Cummins, P. H. Schwartz, D. K. Martin, and W. M. Tierney, "Designing a system for patients controlling providers' access to their electronic health records: Organizational and technical challenges," *J. General Internal Med.*, vol. 30, no. 1, pp. 17–24, 2015.

[2] Microsoft. *Microsoft HealthVault*. [Online]. Available: http://www.healthvault.com, accessed May 1, 2015.

[3] Google Inc. *Google Health*. [Online]. Available: https://www.google.com/health, accessed Jan. 1, 2013.

[4] D. Boneh, G. Di Crescenzo, R. Ostrovsky, and G. Persiano, "Public key encryption with keyword search," in *Proc. EUROCRYPT*, vol. 3027. Interlaken, Switzerland, May 2004, pp. 506–522.

[5] Q. Tang, "Public key encryption schemes supporting equality test with authorisation of different granularity," *Int. J. Appl. Cryptogr.*, vol. 2, no. 4, pp. 304–321, 2012.

[6] P. Liu, J. Wang, H. Ma, and H. Nie, "Efficient verifiable public key encryption with keyword search based on KP-ABE," in *Proc. IEEE 9th Int. Conf. Broadband Wireless Comput., Commun. Appl. (BWCCA)*, Nov. 2014, pp. 584–589.

[7] L. Fang, W. Susilo, C. Ge, and J. Wang, "Public key encryption with keyword search secure against keyword guessing attacks without random oracle," *Inf. Sci.*, vol. 238, pp. 221–241, Jul. 2013.

[8] M.-S. Hwang, S.-T. Hsu, and C.-C. Lee, "A new public key encryption with conjunctive field keyword search scheme," *Inf. Technol. Control*, vol. 43, no. 3, pp. 277–288, 2014.

[9] D. Boneh and B. Waters, "Conjunctive, subset, and range queries on encrypted data," in *Proc. 4th Theory Cryptogr. Conf.*, vol. 4392. Amsterdam, The Netherlands, Feb. 2007, pp. 535–554.

[10] B. Zhang and F. Zhang, "An efficient public key encryption with conjunctive-subset keywords search," *J. Netw. Comput. Appl.*, vol. 34, no. 1, pp. 262–267, 2011.

[11] J. W. Byun and D. H. Lee, "On a security model of conjunctive keyword search over encrypted relational database," *J. Syst. Softw.*, vol. 84, no. 8, pp. 1364–1372, 2011.

[12] M. Ding, F. Gao, Z. Jin, and H. Zhang, "An efficient public key encryption with conjunctive keyword search scheme based on pairings," in *Proc. 3rd IEEE Int. Conf. Netw. Infrastruct. Digit. Content (IC-NIDC)*, Beijing, China, Sep. 2012, pp. 526–530.

[13] J. Shao, Z. Cao, X. Liang, and H. Lin, "Proxy re-encryption with keyword search," *Inf. Sci.*, vol. 180, no. 13, pp. 2576–2587, 2010.

[14] W.-C. Yau, R. C.-W. Phan, S.-H. Heng, and B.-M. Goi, "Proxy re-encryption with keyword search: New definitions and algorithms," in *Proc. Int. Conf. Security Technol.*, vol. 122. Jeju Island, Korea, Dec. 2010, pp. 149–160.

[15] L. Fang, W. Susilo, C. Ge, and J. Wang, "Chosen-ciphertext secure anonymous conditional proxy re-encryption with keyword search," *Theoretical Comput. Sci.*, vol. 462, pp. 39–58, Nov. 2012.

[16] X. A. Wang, X. Huang, X. Yang, L. Liu, and X. Wu, "Further observation on proxy re-encryption with keyword search," *J. Syst. Softw.*, vol. 85, no. 3, pp. 643–654, 2012.

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

[18] M. Bellare, A. Boldyreva, and A. Palacio, "An uninstantiable random-oracle-model scheme for a hybrid-encryption problem," in *Proc. Int. Conf. Theory Appl. Cryptogr. Techn. (EUROCRYPT)*, vol. 3027. Interlaken, Switzerland, May 2004, pp. 171–188.

[19] J. W. Byun, H. S. Rhee, H.-A. Park, and D. H. Lee, "Offline keyword guessing attacks on recent keyword search schemes over encrypted data," in *Proc. 3rd VLDB Workshop Secure Data Manage. (SDM)*, vol. 4165. Seoul, Korea, Sep. 2006, pp. 75–83.

[20] W.-C. Yau, R. C.-W. Phan, S.-H. Heng, and B.-M. Goi, "Keyword guessing attacks on secure searchable public key encryption schemes with a designated tester," *Int. J. Comput. Math.*, vol. 90, no. 12, pp. 2581–2587, 2013.

[21] J. Baek, R. Safavi-Naini, and W. Susilo, "Public key encryption with keyword search revisited," in *Proc. Int. Conf. ICCSA*, vol. 5072. Perugia, Italy, Jun./Jul. 2008, pp. 1249–1259.

[22] L. Guo and W. C. Yau, "Efficient secure-channel free public key encryption with keyword search for EMRs in cloud storage," *J. Med. Syst.*, vol. 39, no. 2, pp. 1–11, 2015.

[23] H. S. Rhee, J. H. Park, W. Susilo, and D. H. Lee, "Trapdoor security in a searchable public-key encryption scheme with a designated tester," *J. Syst. Softw.*, vol. 83, no. 5, pp. 763–771, 2010.

[24] C. Hu and P. Liu, "A secure searchable public key encryption scheme with a designated tester against keyword guessing attacks and its extension," in *Proc. Int. Conf. Adv. Comput. Sci., Environ., Ecoinform., Edu. (CSEE)*, vol. 512. Wuhan, China, Aug. 2011, pp. 131–136.

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

[26] H. S. Rhee, J. H. Park, and D. H. Lee, "Generic construction of designated tester public-key encryption with keyword search," *Inf. Sci.*, vol. 205, pp. 93–109, Nov. 2012.

[27] W.-C. Yau, R. C.-W. Phan, S.-H. Heng, and B.-M. Goi, "Security models for delegated keyword searching within encrypted contents," *J. Internet Services Appl.*, vol. 3, no. 2, pp. 233–241, 2012.

[28] K. Emura, A. Miyaji, and K. Omote, "A timed-release proxy re-encryption scheme," *IEICE Trans. Fundam. Electron., Commun. Comput. Sci.*, vol. 94, no. 8, pp. 1682–1695, 2011.

[29] Q. Liu, G. Wang, and J. Wu, "Time-based proxy re-encryption scheme for secure data sharing in a cloud environment," *Inf. Sci.*, vol. 258, pp. 355–370, Feb. 2014.

[30] K. Liang, Q. Huang, R. Schlegel, D. S. Wong, and C. Tang, "A conditional proxy broadcast re-encryption scheme supporting timed-release," In *Information Security Practice and Experience*. Berlin, Germany: Springer, 2013, pp. 132–146.

[31] J. Li, Y. Shi, and Y. Zhang, "Searchable ciphertext-policy attribute-based encryption with revocation in cloud storage," *Int. J. Commun. Syst.*, doi: 10.1002/dac.2942, 2015.

[32] B. Lynn. *The PBC Library*. [Online]. Available: http://crypto.stanford.edu/pbc/, accessed May 1, 2015.

[33] D. Cash, S. Jarecki, C. Jutla, H. Krawczyk, M.-C. Roşu, and M. Steiner, "Highly-scalable searchable symmetric encryption with support for Boolean queries," in *Advances in Cryptology*, Berlin, Germany: Springer, 2013, pp. 353–373.

[34] D. Cash *et al.*, "Dynamic searchable encryption in very-large databases: Data structures and implementation," in *Proc. Netw. Distrib. Syst. Security Symp. (NDSS)*, Feb. 2014, pp. 1–32.

[35] S. Jarecki, C. Jutla, H. Krawczyk, M. Rosu, and M. Steiner, "Outsourced symmetric private information retrieval," in *Proc. ACM SIGSAC Conf. Comput. Commun. Security*, 2013, pp. 875–888.

Yang Yang received the Ph.D. degree from Xidian University, in 2011. She is with the School of Mathematics and Computer Science, Fuzhou University. Her research interests are in the area of information security and privacy.

Maode Ma received the Ph.D. degree from the Hong Kong University of Science and Technology, in 1999. He is an Associate Professor with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore. His research interests include wireless networks, cloud computing, network security, and privacy.