JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 36, 879-894 (2020)
DOI: 10.6688/JISE.202007_36(4).0012

Authorized Certificateless Conjunctive Keyword Search on Encrypted EHRs from WSNs

YUAN SU[1], YANPING LI[1,+], QIANG CAO[2] AND ZHENQIANG WU[3,4]
[1]*School of Mathematics and Information Science*
[3]*School of Computer Science*
Shaanxi Normal University
Xi'an, 710119 P.R. China
[2]*School of Cyberspace Security*
Beijing University of Posts and Telecommunications
Beijing, 100876 P.R. China
[4]*Guizhou Provincial Key Laboratory of Public Big Data*
Guizhou University
Guiyang, 550025 P.R. China
E-mail: {suyuan; lyp; Cq; zqiangwu}@snnu.edu.cn

Nowadays, mobile wearable sensor devices are increasingly used to collect real-time electronic health records (EHRs), which are encrypted to protect the user privacy and outsourced to the cloud for alleviating the local storage pressure. Unfortunately, encryption will cause the difficulty for the medical institutions to search the target EHRs. To address this challenge, we propose an authorized certificateless conjunctive keyword search on encrypted EHRs. First, our scheme subtly integrates certificateless public key cryptosystem with attribute-based keyword search, which eliminates key escrow problems and provides search permission control. That is to say, only medical institutions specified by the data owners can search and access EHRs in the cloud. Second, our scheme supports conjunctive keyword search to improve search accuracy, and adopts hidden access structure to protect the privacy of users and EHRs. Third, our scheme supports EHRs dynamic updating which enables the data owners to flexibly insert and delete the EHRs in the cloud. Finally, the performance evaluation demonstrates that our scheme is efficient and practical.

Keywords: wireless sensor networks, intelligent medical, conjunctive keyword, data dynamics, electronic health records

1. INTRODUCTION

Based on wireless sensor networks (WSNs), smart medical devices (such as wearable healthcare devices) are widely used to collect the real-time electronic health records (EHRs) of users [1, 2]. To obtain professional health reports, the users can upload their EHRs to the Medical Cloud Service Provider (MCSP), and allow authorized medical institutions to access specific EHRs for analysis. However, EHRs are sensitive data and vulnerable to potential attacks by companies who could make profits from these private data [3]. Hence, the privacy and search permission control of EHRs raise widely concerns in WSNs-based intelligent medical.

Received August 31, 2019; revised October 29 & December 9, 2019; accepted December 10, 2019.
Communicated by Xiaohong Jiang.
+ Corresponding author: lyp@snnu.edu.cn

Generally, encryption is an effective way to solve above problems. However, if the EHRs are encrypted and stored in MCSP, medical institutions are difficult to search the target EHRs from the MCSP unless they download the whole encrypted data in cloud and then decrypt them, which always leads to a huge waste of computation and bandwidth, and low search efficiency. To efficiently search the desired encrypted EHRs, the searchable encryption technique was introduced, which allow the medical institutions to search the target EHRs securely and effectively according to certain trapdoors or tokens. And the attributed-based keyword search can provide more fine-grained access control, which better protects the security of the EHRs. Hence, we attempt to design an attribute-based conjunctive keyword search scheme so that medical institutions can quickly obtain the required EHRs from the medical cloud MCSP.

1.1 Related Work

Recently, searchable encryption (SE) has been studied as one of the research hotpots in cloud storage. Song *et al*. [4] proposed the first symmetric searchable encryption (SSE) scheme, which is inefficient because the search time increases linearly with the size of data collection. Afterwards, many works have added various functions to SSE [5-8], such as dynamic updating, verifiable and multi-level access. The main advantage of these schemes is their higher efficiency without any costly public key operations (such as bilinear pairing, exponentiation). Unfortunately, these works hardly meet the security and privacy requirements in cloud computing environments.

Boneh *et al*. [9] constructed a public key SE (PKSE) scheme with a stronger security model, many new PKSE scheme with different functions have been proposed [10-13], such as conjunctive keyword search, dynamic updating and verification. NI *et al*. [14] proposed a certificate-based keyword SE scheme with specified data dele-tion. Unfortunately, one drawback of this kind of scheme is that any data user (DU) can search and access the data, and search permission flooding brings security risks.

To solve above problem, Sahai *et al*. [15] firstly proposed a fuzzy identity-based encryption scheme which grants the users who have specified pre-defined attributes to decrypt/access data and pioneers fine-gained access control. Later, the key-policy attribute-based encryption (KP-ABE) [16] and the ciphertext-policy attribute-based en-cryption (CP-ABE) [17] are put forward respectively. Afterwards, various SE schemes [12, 13, 18-21, 22] based on CP-ABE or KP-ABE have been proposed. Almost all attribute-based SE schemes use attributes to achieve fine-grained search permission. The data owner (DO) specifies that DU with certain attributes can search their own EHRs [12, 13, 15-19, 23, 24], *i.e.*, the patients allow the medical institutions cooperating with them to search their EHRs. This kind of scheme does not require redundant validation for public key because the DU's attributes are exactly their public keys. Although it has lightened the certificate management issue, the key escrow problem arises in such schemes since the key gener-ation center (KGC) can access all users' private keys.

To overcome above deficiency, Zheng *et al*. [20] first introduced keyword-based SE scheme in certificateless cryptosystem (CLKS) to eliminate the public key certificate and key escrow problem. However, this scheme only supports single keyword search and does not achieve fine-grained search permission. Afterwards, several CLKS works are proposed in [21, 22, 25]. Unfortunately, these works fail to achieve fine-grained search permission. In this paper, we subtly integrate attribute-based keyword searchable encryption scheme with certificateless cryptosystem, and propose a certificateless attribute-based conjunctive keyword SE scheme on encrypted EHRs, which simultaneously achieves fine-grained search permission and highly accurate search results.

1.2 Our Contributions

An authorized certificateless conjunctive keyword search on encrypted EHRs for WSNs-based medical cloud is presented in this work. The contributions are given as follows.

- Our proposed scheme subtly integrates certificateless cryptosystem with attributed-based keyword search, which simultaneously solves the certificate management and key escrow problems, and achieves better fine-grained search, *i.e.*, only medical institutions specified by the DO can search and access his/her EHRs in the cloud.
- Our proposed scheme achieves conjunctive keywords search with higher search accuracy. Based on hidden access structure and authorization, it also provides higher privacy protection for EHRs and users, *i.e.*, the MCSP cannot learn any information of DU's attributes and the content of queried keywords.
- A cuckoo filter is used to build the index, which can improve search efficiency and allow data owners to flexibly manage (insert and delete) their EHRs in the cloud.
- Detailed comparisons between our proposed scheme and the existing state-of-the-art SE schemes in the aspect of the storage and computation costs are given, and the results demonstrated that our scheme is efficient and feasible.

The rest of this paper is organized as follows. Section 2 describes preliminary works, the system model and security model. Our scheme is proposed in Section 3. In Section 4, the performance evaluations of our scheme are given, and the comparisons on computation and storage costs between the related schemes and our scheme are shown. Finally, Section 5 concludes this paper. The appendix gives the security proof of this scheme.

2. PRELIMINARIES AND PROBLEM STATEMENT

2.1 Preliminaries

(A) Bilinear Pairing

Let G_1, G_2 and G_T be three multiplicative cyclic groups with the prime order q, and g_1, g_2 be the generator of group G_1, G_2, respectively. A map $e: G_1 \times G_2 \rightarrow G_T$ is called a bilinear map with following properties:

(1) Bilinear: $\forall g_1 \in G_1, g_2 \in G_2, a, b \in Z_q^*, e(g_1^a, g_2^b) = e(g_1, g_2)^{ab}$;
(2) Non-degenerate: There exista $g_1 \in G_1, g_2 \in G_2$, such that $e(g_1, g_2) \neq 1$;
(3) Computable: $\forall g_1 \in G_1, g_2 \in G_2, e(g_1, g_2) \in G_T$ can be computed efficiently.

(B) Access Structure

In our scheme, we use And-gate to attach different attributes and Or-gate to different values of the same attribute. Let $Att = \{att_1, att_2, \cdots, att_n\}$ be a set of attributes. For each $att_i = \{v_{i1}, v_{i2}, \cdots, v_{in}\}$ is a set of possible values, where $n_i = |att_i|$. $L = \{L_1, L_2, \cdots, L_n\}$ is an attribute list of a DU and $W = \{W_1, W_2, \cdots, W_n\}$ is an access control policy preset by a DO where $L_i \in att_i, W_i \in att_i$. We say the attribute list L satisfies the access control policy W if and only if $L_i \in W_i$ for $1 \leq i \leq n$.

(C) Dynamically Adjustable-capacity Cuckoo Filter

A dynamically adjustable-capacity cuckoo filter (DACF) [26] is a compact vari-ant of cuckoo hashing, which is used to check whether an element belongs to a set. It not only supports data inserting and deleting operations, but also achieves efficient search operation with less space overheads. The general cuckoo filter cannot avoid false positive ratio, since the frequent insertions make the space of hash tables in cuckoo filter become smaller and smaller. The DACF in our scheme can make up this deficiency in general cuckoo filter by allocating a double-sized hash table when the original vacant space is too small to allocate space for a new item or an item to be kicked out. Insertion and deletion can be easily operated over DACF.

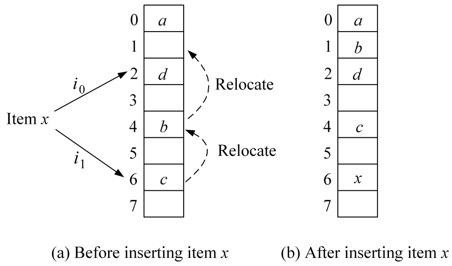


Fig. 1. Illustration of the DACF.

Now combined Fig. 1, we show how the MCSP executes these operations. For the data insertion, if a new item x will be inserted into the hash table, DO computes the fingerprint $fp \leftarrow fingerprint(x)$ and sends the fp to the MCSP. The MCSP computes the position of two alternative candidate buckets $i_0 = hash(fp), i_1 = hash(\bar{fp})$, where is $fingerprint(\cdot)$ a hash function and $\bar{fp}$ denotes the complement of fp. Then it puts the fingerprint fp into the alternative bucket, if both alternative buckets are not empty, it randomly chooses one of them and kicks out the existing item fp_4, inserts fp into the chosen bucket, then reinserts fp_4 by the same insert operation of x, as shown in Fig. 1. If there is still an item to be kicked out after a maximum number of substitutions, it allocates double size of space for a new hash table and rehashes all the fingerprints in the original hash tables into the new hash tables. After that it inserts the item into the new hash tables and discards the original one. When deleting an item x, DO sends the fingerprint of x to MCSP, the MCSP looks up and removes the fingerprint of x from the corresponding candidate bucket if one of the existing fingerprints in two buckets matches the x's fingerprint.

2.2 System Model

Fig. 2 gives the system model of our scheme. There mainly are three types of entities: data owner (DO), a medical cloud service provider (MCSP) and the data user (DU) such as medical institution. A DO wears smart healthcare devices to collect EHRs and uploads the EHRs to the MCSP. The MCSP provides data storage for DO and search services for DU. A DU is the medical institution who search and download specific EHRs from the MCSP for medical uses.

2.3 Security Model and Security Assumption

Generally suppose the MCSP is honest-but-curious, *i.e.*, it honestly executes the pre-defined protocols while attempting to learn as much private information as possible. Assume that the MCSP and DU cannot collude together. There are two types of adversary usually considered in certificateless cryptosystem: Type-I adversary $\mathscr{A}_I$ simulates the outside attacker, who is allowed to replace DO's public key without accessing the system master key. And Type-II adversary $\mathscr{A}_{II}$ models the inside attacker, who can get the system master key without performing public-key substitution attack.

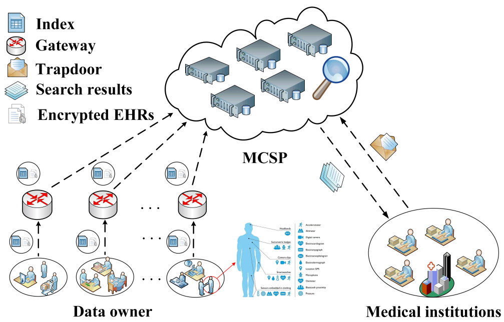


Fig. 2. System model.

Definition 1 (Decisional Linear (DL) Assumption)**.** *Given $v, h, f, R \in G_1$ and v^{r_2}, h^{r_1} where $r_1, r_2 \in Z_q^*$ are unknown, the DL assumption states that any probabilistic polynomial time algorithm $\mathscr{A}$ can determine whether $R = f^{r_1+r_2}$ or not at most with a negligible advantage with respect to the security parameter λ, where the advantage is defined as*

$$Adv_{DL}(\lambda) = |Pr[\mathscr{A}(v,h,v^{r_2},h^{r_1},f^{r_1+r_2})] - Pr[\mathscr{A}(v,h,v^{r_2},h^{r_1},R] = 1|.$$

3. OUR PROPOSEDSCHEME

In this section, an authorized conjunctive keyword search on mobile encrypted EHRs from WSNs is proposed, which is composed of four phases (System setup, EHRs upload, EHRs search and EHRs updating) and eight algorithms, as shown in Fig. 3. Firstly, the key generate center (KGC) publishes system public parameters, and then DO, DU and the MCSP register on the system and get their public/secret keys, respectively. Secondly, according to the system keyword dictionary, a DO extracts keywords from the EHRs to build the index and encrypt EHRs. Thirdly, the gateway of WSNs will collect the index and encrypted EHRs, then upload them to the MCSP. Finally, when searching for specified EHRs, a DU generates a qualified search trapdoor and sends it to MCSP. Then the MCSP conducts the keyword search and returns the desired EHRs to the DU. Finally, the EHRs updating is optionally triggered if the insertion and deletion is actually needed.

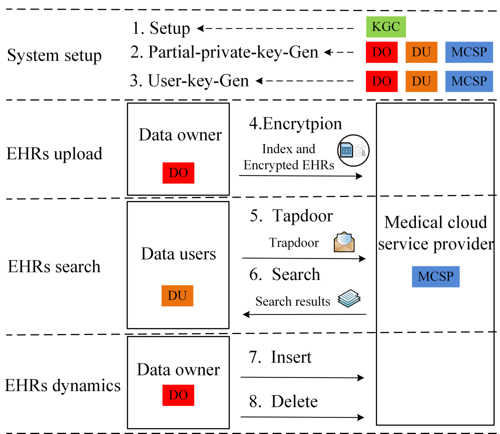


Fig. 3. Framework of our scheme.

A few works [13, 20-22, 25] combine keyword search with certificateless cryptosystem to eliminate certificate management and key escrow. And only the scheme in [13] and our proposed scheme further presented attribute-based keyword search to achieve search permission control, *i.e.*, data owners can preset their EHRs can be searched by medical institutions with specific attributes. We also introduce the conjunctive keyword search to make search results more accurate, and hide the access structure to protect the users' attribute privacy. Moreover, a DACF is used to store keyword index which supports high lookup performance and flexible EHRs updating operations.

3.1 System Setup

In this phase, the KGC mainly generates system public parameters and assists the DO, DU and the MCSP to generate their public/secret keys.

Setup $(1^{\lambda}, Att)$: Given a security parameter λ, $Att = \{att_1, \cdots, att_n\}$ is a set of attributes predefined by the system, where $att_i = \{v_{i1}, \cdots, v_{in}\}$. KGC generates system public parameter pm and the master secret key msk as follows:

(1) Select three multiplicative groups G_1, G_2, G_T of prime order q and the generators g_1, g_2 of group G_1 and G_2, respectively. Let be a bilinear pairing, and are resistant-collision hash functions.

(2) Select $r \in Z_q^*$ and compute $f_A = g_1^r, f_B = g_2^r$.

(3) Choose $a_{ij} \in Z_q^*$ and compute $A_{ij} = g_2^{a_{ij}}$ for each $v_{ij} \in att_i (1 \leq j \leq n_i, 1 \leq i \leq i)$.

(4) Publish the public parameters $pm = \{G_1, G_2, G_T, q, e, g_1, g_2, f_B, \{A_{ij}\}_{(1 \leq j \leq n_i, 1 \leq i \leq n)}, H_1, H_2\}$ and take $msk = \{f_A, \{a_{ij}\}_{(1 \leq j \leq n_i, 1 \leq i \leq n)}\}$ as the master secret key.

Partial-private-key-Gen (pm, msk, L): For any user (DO or DU) with attribute $L = \{L_1, \cdots, L_n\}$, KGC randomly selects value $a \in Z_q^*$ and computes $d_1 = f_A \cdot \prod_{v_{ij} \in L} g_1^{a_{ij}}$, returns the user with partial private key $psk = (a, d_1)$. For the MCSP, the KGC randomly and secretly chooses $a' \in Z_q^*$, and then returns it to the MCSP.

User-key-Gen (pm, psk): The user (DO or DU) randomly selects $b \in Z_q^*$ and computes $d_2 = d_1^{ab}$, and the private key is $sk_U = (a, b, d_2)$, the public key is $pk_U = (f_B \cdot \prod_{v_{ij} \in L} g_1^{a_{ij}})^b$. The MCSP chooses a random value $b' \in Z_q^*$, and generates the public key $pk_{cs} = (cs_1, cs_2) = (g_1^{a'b'}, g_2^{a'b'})$.

Algorithm 1

Require: The public parameter pm; The public key of MCSP pk_{cs}; The access policy W. The EHRs F; The keywords set kw_{ext} extracted from F;
Ensure: The ciphertexts C; The index $\mathbb{CF}$;

```
Set access policy W = {W_1, W_2, ..., W_n};
Select t ∈ Z_q^* and compute I_0 = (f_B · ∏_{v_ij ∈ W} A_ij)^t, I_2 = cs_1^t, set I = {I_0, I_2};
for i = 1 to k do
    Compute I_{1,i} = g_2^{t H_2(kw_i)}, fp_i = H_1(kw_i);
end for
for i = 1 to N do
    Compute ciphertext c_i for by symmetric encryption algorithm (AES) and generate
    the pointer p_i pointing c_i;
end for
Compose C = {c_1, ..., c_N};
for j = to k do
    for i = 1 to N do
```

12: **if** $kw_i \in f_i$ **then**
13: add the pointer p_i to the set P_j;
14: **end if**
15: **end for**
16: **end for**
17: Set $CF_i = \{fp_i, P_i, Aux\}$ and $\mathbb{CF} = \{CF_i\}_{1 \le i \le k}$ where $Aux = \{I_{1,i}, I\}$;
18: Return $\mathbb{CF}, C$;

3.2 EHRs Upload

In this phase, a DO uploads encrypted EHRs C and an index $\mathbb{CF}$ to the MCSP.

Encryption $(pm, W, F, kw_{ext}, pk_{cs})$: When uploading the EHRs $F = \{f_1, \cdots, f_N\}$ with keywords $kw_{ext} = \{kw_1, \cdots, kw_k\}$, a DO performs Algorithm 1 and finally gets the index $\mathbb{CF}$ and the ciphertexts C of F. In Algorithm 1, lines 1-3 mainly generate a hidden fine-grained access control policy which is used to protect user's attributes privacy, and lines4-16 are used to build the index $\mathbb{CF}$.

It must be noticed that we store three parts into the candidate bucket of the index $\mathbb{CF}$: fingerprint fp_i, pointer set P_i and the auxiliary information $Aux = \{I_{1,i}, I\}$, which is slightly different from the descriptions in Section 2.5, where only a single fingerprint is stored in each candidate bucket. A dynamically adjustable-capacity cuckoo filter (DACF) are used to build an index, which supports the fast search since only two hash computations are required to locate the file containing the queried keywords.

3.3 EHRs Search

In this phase, a DU generates queried trapdoor, then the MCSP searches and returns EHRs that satisfies DUs' queried trapdoor.

Trapdoor $(pm, pk_{cs}, sk_U, kw_{query})$: For a set kw_{query} including s queried keywords, a DU computes $T_0 = (g_1^{ab} \cdot cs_1)^p$, $T_2 = pb^{-1}$, the fingerprint $fp_i = H_1(kw_i)$ and $T_{1,i} = (d_2)^{pH_2(kw_i)^{-1}}$ for each $kw_i \in kw_{query} (1 \le i \le s)$, where p is randomly chosen from Z_q^*. Finally, DU sends the queried trapdoor $Trap = \{T_0, T_2, \{T_{1,i}, fp_i\}_{i=1}^{i=s}\}$ to the MCSP.

Search $(pm, Trap, pk_U)$: Upon receiving the trapdoor $Trap$ from the DU, the MCSP performs Algorithm 2 and obtains the search results R. Finally, the MCSP responds the DU with R.

Algorithm 2

Require: The public parameter pm; The search trapdoor $Trap$; The public key of DU pk_U;
Ensure: The search result R;
for $j = 1$ to s **do**
the MCSP looks up fp_j in the $\mathbb{CF}$, if fp_j exists, then gets the auxiliary information $Aux = \{I_{1,j}, I\}$, otherwise returns $R = \emptyset$;
if

$$e(I_0, T_0) = e(I_{1,j}, T_{1,j}) e(I_2, (pk_U)^{T_2}) \tag{1}$$

then
add the pointer P_j to the set CR;
end if
end for

We get $CR = P_1, P_2, \cdots, P_s$. Then we perform the set intersection on CR (*i.e.*, $P_{SR} = \cap P_i, P_i \in CR, 1 \leq i \leq s$) to get the target pointers set P_{SR}, according which the MCSP extracts ciphertexts to the set R;
Return R.

Fast location: Here we display how the MCSP efficiently locate the fingerprint fp in the index $\mathbb{CF}$. The MCSP only need to computes two hash values: $i_0 = hash(fp), i_1 = hash(\bar{fp}) \oplus i_0$, the fingerprint fp must be in the corresponding bucket of i_0 or i_1.

When DU obtains the desired encrypted EHRs, he may identify himself to DO and asks for the decryption key, which guarantees absolute control of DO over the private EHRs. And decryption is beyond the research scope of our scheme.

3.4 EHRs Updating

Considering that EHRs are collected by the mobile sensor devices, it is unreasonable for users to modify EHRs at will, which will greatly decrease the credibility of EHRs. Therefore, our proposed scheme only allows DO to dynamically insert and delete the EHRs. The deletion facilitates DO to manage their own EHRs, *i.e.*, DO not only can share their personal EHRs for diagnosis or medical research, but also flexibly delete their own EHRs for protecting their privacy.

Insert: Suppose a DO wants to insert a supplementary EHR f with keywords $kw_{ins} = \{kw_1, kw_2, \cdots, kw_{k'}\}$ to $\mathbb{CF}$. He first computes the ciphertext c of f and fingerprint $\{fp_i\}_{1 \leq i \leq k'}$ of kw_{ins}. Then, he generates a pointer p_f pointing to c, and sends insert information $I_f = \{c, p_f, \{fp_i\}_{i=1}^{k'}\}$ to the MCSP. Upon receiving I_f, for each fingerprint $fp_i (1 \leq i \leq k')$, the MCSP lookups fp_i in the index $\mathbb{CF}$ and then inserts the pointer p_f to the pointer set P_i of the bucket CF_i. Finally, the MCSP store the ciphertext c.

Delete: Suppose a DO wants to delete an EHR f with keywords $kw_{del} = \{kw_1, \cdots, kw_{k''}\}$. He first computes the ciphertext c of EHR f and the fingerprint $\{fp_i\}_{1 \leq i \leq k''}$ of kw_{del}. Then he sends delete information $D_f = \{c, \{fp_i\}_{i=1}^{i=k''}\}$ to the MCSP. Upon receiving D_f, the MCSP finds the pointer p_d which points to the ciphertext c, then the MCSP lookups each fingerprint $fp_i (1 \leq i \leq k'')$ in the $\mathbb{CF}$ and deletes the pointer p_d from the pointer set P_i in bucket CF_i. Finally, the MCSP deletes ciphertext c.

4. SECURITY AND PERFORMANCE ANALYSES

4.1 Privacy Protection and Correctness

The privacy of our proposed scheme includes attributes privacy, EHRs privacy, and index privacy.

Attributes privacy is much more important than it might seem since attributes includes some sensitive information of DUs. We hide the access control policy and embed the attributes into the public/secret keys, which not only protects attributes privacy, but also facilitates a DO to authenticate DUs without leaking their attributes to the MCSP. Thus, our proposed scheme better protects the attributes privacy.

EHRs privacy consists of EHRs and keywords contained in EHRs, encryption guarantees that no one else can obtain the content of EHRs, and only the authorized DU can search/access the EHRs.We utilize the fingerprints of keywords contained in the EHRs to build an index and support efficiently search. Hence the MCSP and unauthorized DU cannot learn any useful information from the fingerprints of keywords.

As for an index $\mathbb{CF}$, the content of the bucketCF_i is $\{fp_i, P_i, Aux\}$ which do not leak sensitive information to the MCSP. Because $fp_i = H_1(kw_i)$ is hash value and H_1 is a resistant-collision hash function, P_i is a pointer set and $Aux = I_{1,i}, I$ is the hidden access structure, the MCSP is blind to all of them and does not get any useful content.

Consequently, our scheme protects the attributes privacy of DUs, EHRs privacy, and index privacy.

Then the correctness of Eq. (1) in Algorithm 2 which is responsible for searching accurate EHRs can be verified as follows:

The left of Eq. (1) is

$$\begin{aligned} e\Big(I_0, T_0\Big) &= e\Big((f_B \cdot \textstyle\prod_{v_{ij}\in W} A_{ij})^t, (g_1^{ab} \cdot g_1^{a'b'})^p\Big) \\ &= e\Big(g_2^{t(r+\sum_{v_{ij}\in W} a_{ij})}, g_1^{p(a'b'+ab)}\Big) \\ &= e\Big(g_1, g_2\Big)^{pt(a'b'+ab)(r+\sum_{v_{ij}\in W} a_{ij})}. \end{aligned}$$

The right of Eq. (1) is

$$\begin{aligned} &e\Big(I_{1,j}, T_{1,j}\Big) \cdot e\Big(I_2, (pk_U)^{T_2}\Big) \\ &= e\Big(g_2^{tH_2(kw'_j)}, g_1^{pab(r+\sum_{v_{ij}\in L} a_{ij})H_2(kw_j)^{-1}}\Big) \cdot e\Big(g_1^{a'b't}, (f_B \cdot \textstyle\prod_{v_{ij}\in L} A_{ij})^p\Big) \\ &= e\Big(g_2^t, g_1^{pab(r+\sum_{v_{ij}\in L} a_{ij})\cdot H_2(kw'_j)\cdot H_2(kw_j)^{-1}}\Big) \cdot e\Big(g_1^{a'b't}, g_2^{p(r+\sum_{v_{ij}\in L} a_{ij})}\Big) \\ &= e\Big(g_2, g_1\Big)^{pt(ab\cdot H_2(kw'_j)\cdot H_2(kw_j)^{-1}+a'b')((r+\sum_{v_{ij}\in L} a_{ij})} \end{aligned}$$

Eq. (1)$e(I_0, T_0) = e(I_{1,j}, T_{1,j})e(I_2, (pk_U)^{Q_2})$ holds if and only if $\sum_{v_{ij}\in L} a_{ij} = \sum_{v_{ij}\in W} a_{ij}$ and $kw'_j = kw_j$. $\sum_{v_{ij}\in L} a_{ij} = \sum_{v_{ij}\in W} a_{ij}$ means DU's attributes L satisfies the access control policy W specified by the DO (*i.e.*, $L_i \in W_i$, $1 \le i \le n$, where $L_i \in L$, $W_i \in W$) and $kw'_j = kw_j$ means the queried keywords are exactly contained in the search EHRs. Thus, Eq. (1) holds when the searched EHRs contained the queried keyword and the attributes of DU satisfies the access policy.

4.2 Security Proof

The security of our proposed scheme can be assured by the Theorem 1, the proofs are shown in the Appendix.

Theorem 1. *Let Adv_{H_2} be the advantage of $\mathscr{A}$ breaking the collision-resistant hash function H_2 (i.e., $Adv_{H_2} = |Pr[(H_2(kw_0) = H_2(kw_1)) \cap (kw_0 \ne kw_1) | kw_0, kw_1 \in \{0,1\}^*]|$) and $Adv_{DL}(\lambda)$ be the advantage of $\mathscr{A}$ breaking the DL assumption, then the advantageof $\mathscr{A}_I$ and $\mathscr{A}_{II}$breaking the keyword indistinguishability is $Adv_{\mathscr{A}_I,\mathscr{A}_{II}} \le Adv_{DL}(\lambda) + Adv_{H_2}$.*

We compare our scheme with several works in terms of security, as shown in Table 1. All schemes except the scheme in [4] can resist key guessing attack to protect the keyword privacy from the adversary. Furthermore, our scheme and the scheme in [4, 11, 12, 19] better hide the access control policy to protect the attribute privacy. Part of schemes achieve search permission control which allows DU specified by DO can search/access EHRs in the cloud. Part of schemes achieve different levels of privacy protection such as attributes privacy, EHRs privacy and index privacy. Overall, our proposed scheme has good comprehensive security.

Table 1. Security comparisons.

Scheme	[5]	[11]	[12]	[19]	[20]	[21]	[22]	[25]	Ours
Keyword guessing attack		✓	✓	✓	✓	✓	✓	✓	✓
Hidden access control	✓	✓	✓	✓					✓
Search permission control	✓			✓		✓		✓	✓
Privacy protection	✓		✓	✓					✓

4.3 Performance Analyses

The notations used in the following are given in Table 2. We compare our scheme with the latest certificateless SE schemes on the storage and computation costs, as shown in Tables 3-6. We do all experiments by using GNU Multiple Arithmetic (GMP) library and Pairing Based Cryptography and run at Windows 7 with a 2.60 GHz Intel Core i5-4210M and 4 GB memory. Each experiment is repeated 50 times to determine the average execution time. The simulation analyses are presented in Figs. 4 and 5.

Table 2. Notations.

Notation	Meaning	Notation	Meaning
$\|G_{1/2/T}\|$	Element size in $G_{1/2/T}$	$\|G_T\|$	Element size in G_T
k	The number of keywords in F	$\|Z_q\|$	Element size in Z_q
l	Bit-length of a hash value G_T	H	Hash operation
Mu	Multiplication in Z_q	Ad	Addition in Z_q
PM	Point multiplication in G_1	In	Multiplication inverse in Z_q
$\|Exp_{1/2/T}\|$	Exponentiation in $Exp_{1/2/T}$	Add	Addition in G_1
Mul	Multiplication in G_1	P	Pairing operation
$\|W\|$	Number of attributes in W	s	Number of queried keyword

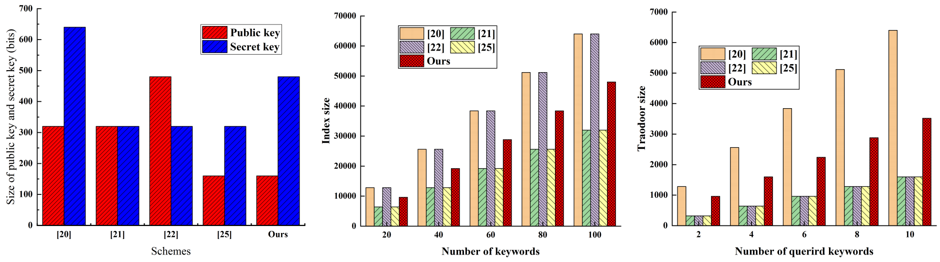
4.3.1 Storage costs

Storage costs mainly include the size of public/private keys, the size of index and trapdoor. Generally suppose $|G_1| = |G_2| = |Z_q| = 160$ bits and several certificateless key word search schemes are considered, the results are shown in Table 3 and Fig. 4.

Table 3. Comparisons of storage costs.

Scheme	Public key	Secret Key	Index	Trapdoor
[20]	$2\|G_1\|$	$2(\|G_1\|+\|Z_q\|)$	$4k\|G_1\|$	$4s\|G_1\|$
[21]	$2\|G_1\|$	$2\|Z_q\|$	$2k\|G_1\|$	$s\|G_T\|$
[22]	$3\|G_1\|$	$2\|G_1\|$	$2k(\|G_1\|+\|Z_q\|)$	$s\|G_1\|$
[25]	$\|G_1\|$	$\|G_1\|+\|Z_q\|$	$k(\|G_1\|+l)$	$s\|Z_q\|$
Ours	$\|G_2\|$	$\|G_1\|+2\|Z_q\|$	$k(\|G_1\|+\|G_2\|+\|Z_q\|)$	$(s+1)(\|G_1\|+\|Z_q\|)$

From Table 3, the size of public key in our scheme is obviously smaller than the other schemes and the size of secret key also has certain advantage. Suppose the EHRs set F has k keywords, the index size of our scheme is $k(|G_1| + |G_2| + |Z_q|)$ ($480k$ bits), $k|Z_q|$ ($160k$ bits) is the size of fingerprints which are used for fast location, $k(|G_1| + |G_2|)$ ($320k$ bits) is the size of hidden access policy which protects the attribute privacy of DU and supports fine-grained search permission control. Since our scheme provides $s(1 \leq s \leq k)$ conjunctive keyword search, other schemes should perform s times single keyword search and generate s trapdoors to be fair. The trapdoor size of our scheme also is less than that of the scheme in [20], and higher than those of the schemes in [21, 22, 25] since our scheme needs to compute the fingerprints whose size is $s|Z_q|$ to support efficient locate the target EHRs while other schemes do not refer to the detailed process of locating the target files. The comparison results are shown in Fig. 4, which is consistent with the theoretical analysis in Table 3.



(a) Public and secret key size. (b) Index size. (c) Trapdoor size.

Fig. 4. Comparisons of communication costs.

4.3.2 Computation costs

We assess the computation (time) costs of Encryption algorithm, Trapdoor algorithm and Search algorithm, as shown in Fig. 5, Tables 4-6.

Table 4. Computation costs comparisons of Encryption algorithm.

Scheme	Encryption	Time costs ($k=100$)
[20]	$k(2H+4Mu+2Ad+Mul+4Exp_1)$	821.50 ms
[21]	$k(2H+Mu+Ad+2PM+2Add)$	328.22 ms
[22]	$k(5H+2Mu+4PM+4Add+2Exp_1)$	3761.2 ms
[25]	$k(5H+Mu+PM+2Add+2Exp_T+2P)$	4564.5 ms
Ours	$2kH+(\lvert W\rvert+1)Mul+kMu+kExp_1+Exp_2$	257.13 ms

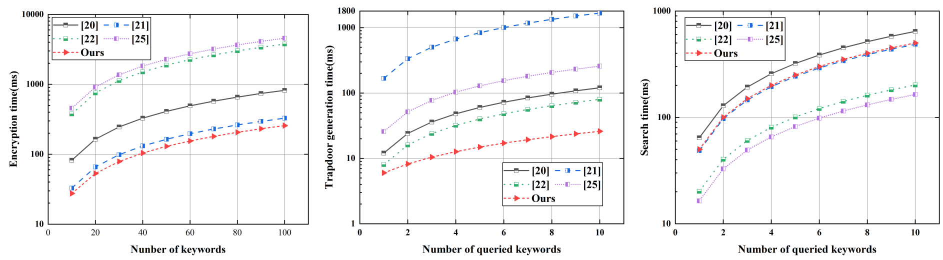
In Encryption process, we consider the computation costs for encrypting EHRs set F with k keywords. From Table 4, we can see our scheme has lower computation cost than other schemes because our scheme does not need the time-consuming pair operations and less exponentiations, and our scheme only needs 257.13 ms when the keywords contained in the EHRs set F is 100. As shown in Fig. 5 (a), our scheme needs less time to encrypt the EHRs when the keywords contained in the EHRs set F varies from 1 to 1000, which is very suitable for practical deployment.

Table 5. Computation costs comparisons of Trapdoor algorithm.

Scheme	Trapdoor	Time costs ($k=10$)
[20]	$s(2H+7Mu+4Ad+Add+2Mul+6Exp_1)$	120.35 ms
[21]	$s(2H+Mu+Ad+2PM+2Add+P)$	1675.8 ms
[22]	$s(Ad+In+PM)$	80.11 ms
[25]	$s(5H+P+PM+Add)$	257.78 ms
Ours	$sH+(s+1)In+Mul+Mu+(s+2)Exp_1$	25.98 ms

In Trapdoor generation, the scheme in [21] is the most computation-consuming since it needs 1675 ms to generate the trapdoor for 10 queried keywords as shown in Table 5 while our scheme needs 25.98 ms, which is the lowest among the five schemes.

For Search process, we do experiments on the total number 1×10^4 of EHRs, the result is shown in Fig. 5 (c). And the computation costs of Search process increase with s, which is consistent with the theoretical study in Table 6. Our scheme requires 500.74 ms to search the desired EHRs with 10 queried keywords, which is acceptable in practice.



(a) Encryption time. (b) Trapdoor generation time. (c) Search time.

Fig. 5. Comparisons of computation costs.

Table 6. Computation costs comparisons of search algorithm.

Scheme	Search	Time costs ($k = 10$)
[20]	$s(Inv + 2Mul + 4P)$	643.23 ms
[21]	$s(H + Mul + 2P + 2PM + 2Add)$	485.63 ms
[22]	$s(H + Mu + In + 2Exp_1 + P)$	201.67 ms
[25]	$s(H + P)$	163.98 ms
Ours	$s(Exp_2 + 3P)$	500.74 ms

5. CONCLUSIONS

In this paper, we propose an authorized certificateless conjunctive keyword search scheme on encrypted EHRs from WSNs. We combine certificateless cryptosystem with attribute-based keyword search, which eliminates the certificate management and key escrow problem, and achieves better fine-grained search permission control. We also adopt conjunctive keyword search to improve search accuracy and hidden access structure to protect the privacy of users and EHRs. A dynamically adjustable-capacity cuckoo filter (DACF) is used to build the index for encrypted EHRs, and supports fast EHRs location and efficient EHRs updating. The experimental simulations demonstrate that the proposed scheme can achieve better comprehensive performance (efficiency and privacy-preserving) in terms of storage and computation costs.

ACKNOWLEDGMENT

This work was partly supported by the National Natural Science Foundation of China (61802243), the Key R&D Program in industry field of Shaanxi Province (2019GY-013), the Fundamental Research Funds for the Central Universities (GK201901008, GK201903011), the Major Scientific and Technological Special Project of Guizhou Province (20183001), the Foundation of Guizhou Provincial Key Laboratory of Public Big Data (2018BDKFJJ004).

REFERENCES

1. E. Rodriguez-Villegas, S. Iranmanesh, and S. A. Imtiaz, "Wearable medical devices: High-level system design considerations and tradeoffs," *IEEE Solid-State Circuits Magazine*, Vol. 10, 2018, pp. 43-52.
2. C. Zhang, H. Cho, and C. Chen, "Emergency-level-based healthcare information offloading over fog network," *Peer-to-Peer Networking and Applications*, 2019, pp. 1-11.
3. S. Zhang, X. Zhang, and X. Ou, "After we knew it: empirical study and modeling of cost-effectiveness of exploiting prevalent known vulnerabilities across iaas cloud," in *Proceedings of the 9th ACM Symposium on Information, Computer and Communications Security*, 2014, pp. 317-328.
4. D. X. Song, D. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," in *Proceeding of IEEE Symposium on Security and Privacy*, 2000, pp. 44-55.
5. H. Li, Y. Yang, Y. Dai, J. Bai, and Y. Xiang, "Achieving secure and efficient dynamic searchable symmetric encryption over medical cloud data," *IEEE Transactions on Cloud Computing*, 2017, doi: 10.1109/TCC.2017.2769645.
6. W. Sun, X. Liu, W. Lou, Y. T. Hou, and H. Li, "Catch you if you lie to me: Efficient verifiable conjunctive keyword search over large dynamic encrypted cloud data," in *Proceedings of IEEE Conference on Computer Communications*, 2015, pp. 2110-2118.
7. J. Alderman, K. M. Martin, and S. L. Renwick, "Multi-level access in searchable symmetric encryption," in *Proceedings of International Conference on Financial Cryptography and Data Security*, 2017, pp. 35-52.
8. L. Qin, X. Nie, X. Liu, P. Tao, and W. Jie, "Verifiable ranked search over dynamic encrypted data in cloud computing," in *Proceedings of IEEE/ACM 25th International Symposium on Quality of Service*, 2017, pp. 1-6.
9. D. Boneh, G. D. Crescenzo, R. Ostrovsky, and G. Persiano, "Public key encryption with keyword search," in *Advances in Cryptology – EUROCRYPT*, 2004, pp. 506-522.
10. Y. Miao, J. Ma, X. Liu, J. Zhang, and Z. Liu, "Vkse-mo: Verifiable keyword search over encrypted data in multi-owner settings," *Science China Information Sciences*, Vol. 60, 2017, pp. 122 105:1-122 105:15.
11. Y. Yang and M. Ma, "Conjunctive keyword search with designated tester and timing enabled proxy re-encryption function for e-health clouds," *IEEE Transactions on Information Forensics and Security*, Vol. 11, 2016, pp. 746-759.
12. W. Sun, S. Yu, W. Lou, Y. T. Hou, and H. Li, "Protecting your right: Attribute-based keyword search with fine-grained owner-enforced search authorization in the cloud," in *Proceedings of IEEE Conference on Computer Communications*, 2014, pp. 226-234.
13. G. Zhang, "Attribute-based certificateless cryptographic system," *Journal of Computers*, Vol. 9, 2014, pp. 72-77.
14. L. Ni and C. Xu, "Dynamic searchable encryption scheme based on identity," *Computer Engineering*, Vol. 45, 2019, pp. 136-140.
15. A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *Proceedings of Annual International Conference on the Theory and Applications of Cryptographic Techniques*, 2005, pp. 457-473.

16. V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *Proceedings of the 13th ACM conference on Computer and Communications Security*, 2006, pp. 89-98.
17. J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *Proceedings of IEEE Symposium on Security and Privacy*, 2007, pp. 321-334.
18. F. L. Ţiplea, C. C. Drăgan, and A.-M. Nica, "Key-policy attribute-based encryption from bilinear maps," in *Proceedings of International Conference for Information Technology and Communications*, 2017, pp. 28-42.
19. Q. Cao, Y. Li, Z. Wu, Y. Miao, and J. Liu, "Privacy-preserving conjunctive keyword search on encrypted data with enhanced fine-grained access control," *World Wide Web*, 2019, Vol. 23, pp. 1-31.
20. Q. Zheng, X. Li, and A. Azgin, "Clks: Certificateless keyword search on encrypted data," in *Proceedings of International Conference on Network and System Security*, 2015, pp. 239-253.
21. D. He, M. Ma, S. Zeadally, N. Kumar, and K. Liang, "Certificateless public key authenticated encryption with keyword search for industrial internet of things," *IEEE Transactions on Industrial Informatics*, Vol. 14, 2017, pp. 3618-3627.
22. R. Elhabob, Y. Zhao, I. Sella, and H. Xiong, "Efficient certificateless public key cryptography with equality test for internet of vehicles," *IEEE Access*, Vol. 7, 2019, pp. 68 957-68 969.
23. K. Emura, A. Miyaji, A. Nomura, K. Omote, and M. Soshi, "A ciphertext-policy attribute-based encryption scheme with constant ciphertext length," in *Proceedings of International Conference on Information Security Practice and Experience*, 2009, pp. 13-23.
24. Y. Fan, X. Wu, and J. Wang, "Multi-authority attribute-based encryption access control scheme with hidden policy and constant length ciphertext for cloud storage," in *Proceedings of IEEE 2nd International Conference on Data Science in Cyberspace*, 2017, pp. 205-212.
25. Y. Lu and J. Li, "Constructing certificateless encryption with keyword search against outside and inside keyword guessing attacks," *China Communications*, Vol. 16, 2019, pp. 156-173.
26. T. Xiang, X. Li, F. Chen, Y. Yang, and S. Zhang, "Achieving verifiable, dynamic and efficient auditing for outsourced database in cloud," *Journal of Parallel and Distributed Computing*, Vol. 112, 2018, pp. 97-107.

A. APPENDIX

Proof. The security games are played between $\mathscr{A}_I$ and a challenger $\mathscr{B}$, who maintains two lists: **Trapdoorlist** T_L and **UserInfoList** U_L. T_L stores the tuple $[L, kw, Trap]$, which means that the search trapdoor $Trap$ with respect to the keyword kw for the attribute L has been queried by $\mathscr{A}_I$. U_L stores the tuple $[L, psk_L, sk_L, pk_L, \mathrm{P}_1, \mathrm{P}_2, \mathrm{P}_3]$ where Boolean value $P_1 = 1$ means that $\mathscr{A}_I$ has acquired and otherwise not, $P_2 = 1$ means that $\mathscr{A}_I$ has acquired and otherwise not, and $P_3 = 1$ means that $\mathscr{A}_I$ has replaced and otherwise not. Let S_i denote the event of $\mathscr{A}_I$ winning the game i and $Adv_{\mathscr{A}_I}$ be the advantage of $\mathscr{A}_I$.

Game 1: Setup: $\mathscr{B}$ runs **Setup**$(1^\lambda, Att t)$ to initialize the system parameter pm and the master key msk. And $\mathscr{B}$ sends pm to $\mathscr{A}_I$ and sets two lists T_L and U_L empty.

Phase 1: $\mathscr{A}_I$ is allowed to query the following oracles in polynomial many times. We use the bracket $[\bullet]$ to indicate the input to the oracle from $\mathscr{A}_I$.

Partial-Private-Key-Gen$[L]$**:** Upon receiving the user attribute list from $\mathscr{A}_I$, the challenger $\mathscr{B}$ returns psk_L if psk_L in U_L is not null. Otherwise, the challenger runs **Partial-Private-Key-Gen**(pm, msk, L)) to get psk_L, adds $[L, psk_L, *, *, 1, 0, 0]$ to U_L where * means null, and returns psk_L to $\mathscr{A}_I$.

User-key-Gen$[L]$**:** Receiving a user's attribute list L from $\mathscr{A}_I$, $\mathscr{B}$ works as follows:

(1) If sk_L and pk_L in U_L is not null, then $\mathscr{B}$ retrieves sk_L.

(2) Else if psk_L in U_L is not null, then $\mathscr{B}$ retrieves psk_L, runs **User-key-Gen** (psk_L) to get sk_L and pk_L. Then $\mathscr{B}$ adds sk_L and pk_L to the U_L.

(3) Otherwise, $\mathscr{B}$ runs **Partial-Private-Key-Gen** (pm, msk, L) to get psk_L and **User-key-Gen**(psk_L) to get sk_L and pk_L. Then $\mathscr{B}$ adds (L, psk_L, sk_L, pk_L) to the U_L.

$\mathscr{B}$ updates $P_1 = 1$ and $P_2 = 1$ in U_L with respect to L and returns sk_L and pk_L to $\mathscr{A}_I$.

Replace-Public-Key$[L, pk_{L'}]$**:** Given the user attribute list L and the replaced public key $pk_{L'}$ from $\mathscr{A}_I$ (assume that $pk_{L'}$ has been generated before), $\mathscr{A}$ updates $pk_{L'}$ in U_L with pk_L, and sets $P_3 = 1$ with respect to L.

Gen-Trapdoor$[L, kw]$: Given the user' attributes list L from $\mathscr{A}_I$, $\mathscr{B}$ retrieves sk_L from U_L, runs **Trapdoor**(pm, pk_{cs}, sk_L, kw) to get trapdoor. $\mathscr{B}$ adds $[L, kw, Trap]$ to the T_L and returns trapdoor to $\mathscr{A}_I$.

Challenge Phase: $\mathscr{A}_I$ presents two keywords kw_0 and kw_1, and the user' attributes list L^*. Let $[L^*, psk_{L^*}, sk_{L^*}, pk_{L^*}, P_1, P_2, P_3]$ be the tuple stored in U_L, we require that $P_1 = 0$ and $P_2 = 0$ which meaning that psk_{L^*} and sk_{L*} are not acquired by $\mathscr{A}_I$ and both $(L^*, kw_0, Trap_0)$ and $(L^*, kw_1, Trap_1)$ are not stored in T_L. $\mathscr{B}$ randomly picks $x \leftarrow \{0, 1\}$, runs **Trapdoor**$(pm, pk_{cs}, sk_L, kw_x)$ to get the trapdoor $Trap$ and returns it to $\mathscr{A}_I$.

Phase 2: $\mathscr{A}_I$ continues to query the oracle as in **Phase 1** and follows these restrictions:

(1) $\mathscr{A}_I$ cannot query **Partial-Private-Key-Gen**$[L^*]$ or **Private-Key-Gen**$[L^*]$.

(2) $\mathscr{A}_I$ cannot query **Trapdoor**$[L^*, kw_0, Trap_0]$ or Trapdoor$[L^*, kw_1, Trap_1]$.

Guess: $\mathscr{A}_I$ outputs a bit x^*. We say $\mathscr{A}_I$ wins the game if $x^* = x$.

Game 2: In this game, $\mathscr{B}$ proceeds with the steps as defined in **Game 1** except that H_2 is set to be a perfect collision-resistant hash function. Note that, **Game 1** is the original attack game , therefore, we have $|Pr[S_1] - Pr[S_2]| = Adv_{H_2}$.

Game 3: $\mathscr{B}$ plays the **Game 3** the same as **Game 1** except for the challenge phase.

$\mathscr{A}_I$ presents two keywords kw_0 and kw_1, and the user' attributes list L^*. Let $[L^*, psk_{L^*}, sk_{L^*}, pk_{L^*}, P_1, P_2, P_3]$ be the tuple stored in U_L, we require that (1) $P_1 = 0$ and $P_2 = 0$ mean that psk_{L^*} and sk_{L^*} are not acquired by $\mathscr{A}_I$. (2) Both $(L^*, kw_0, Trap_0)$ and $(L^*, kw_1, Trap_1)$ are not stored in T_L.

The challenger $\mathscr{B}$ runs **Partial-Private-Key-Gen**$(pm, msk, L*)$ to get psk_{L*}, and $\mathscr{B}$ sets $sk_{L^*} = (d_1 \cdot h^{r_1}, a, b)$ where a, b are randomly chosen from Z_q^*. And given an access policy W, $\mathscr{B}$ computes $I_0 = (f_B \prod_{v_{i,j} \in W} A_{i,j})^t$ and $I_2 = (v^{r_2})^t$, where t is randomly chosen from Z_q^*. Then $\mathscr{B}$ randomly picks $x \leftarrow 0, 1$, runs **Trapdoor**$(pm, pk_{cs}, sk_L, kw_x)$ to get the trapdoor $Trap^*$ where $T_0 = R^p$ and returns $Trap^*$ to $\mathscr{A}_I$.

The distinguishable probability between **Game 2** and **Game 3** is related to the DL problem, then $|Pr[S_2] - Pr[S_3]| \leq Adv_{DL}(\lambda)$. Moreover, as in **Game 3**, kw_x is hidden perfectly, so $Pr[S_3] = \frac{1}{2}$.

This completes the simulation and has the following inequalities:

$$|Pr[S_1] - Pr[S_3]| = |Pr[S_1] - \frac{1}{2}| \leq Adv_{DL}(\lambda) + Adv_{H_2}$$

Therefore, $Adv_{\mathscr{A}_I} = |Pr[S_1] - \frac{1}{2}| \leq Adv_{DL}(\lambda) + Adv_{H_2}$.

The proof for $\mathscr{A}_{II}$ uses similar technologies as above proof, hence we skip it here. Note that $\mathscr{A}_{II}$ will not query the oracle **Partial-Private-Key-Gen** because it has the master key. Therefore, **Theorem 1** has been proven. $\square$

Yuan Su received the B.S. degree from Shaanxi Normal University in 2018. He is currently pursuing the M.S. degree in Computational Mathematics with the School of Mathematics and Information Science, Shaanxi Normal University at Xi'an, China. His research interests include security protocols and its analysis in cloud storage.

Yanping Li received the M.S. degree from Shaanxi Normal University in 2004 and the Ph.D. degree from Xidian University in 2009. She is currently an Associate Professor in the School of Mathematics and Information Science, Shaanxi Normal University at Xi'an, China. Her research interests include public-key cryptography and its applications.

Qiang Cao received the M.S. degree from Shaanxi Normal University in 2019, and now he is persuing the Ph.D. degree in Beijing University of Posts and Telecommunications. His research interests include public-key cryptography, secure multi-party computation and its applications.

Zhenqiang Wu received his Ph.D. degree from Xidian University, China, in 2007. He is currently a Full Professor of Shaanxi Normal University, China. His research interests include computer communications networks, wireless networks, network security, anonymous communication, and privacy protection.