sensors

MDPI

Article

Efficient Privacy-Preserving Access Control Scheme in Electronic Health Records System

Yang Ming * and Tingting Zhang

School of Information Engineering, Chang'an University, Xi'an 710064, China; zhangting141012@163.com
* Correspondence: yangming@chd.edu.cn; Tel.: +86-136-091-16306

Received: 8 September 2018; Accepted: 16 October 2018; Published: 17 October 2018

Abstract: The sharing of electronic health records (EHR) in cloud servers is an increasingly important development that can improve the efficiency of medical systems. However, there are several concerns focusing on the issues of security and privacy in EHR system. The EHR data contains the EHR owner's sensitive personal information, if these data are obtained by a malicious user, it will not only cause the leakage of patient's privacy, but also affect the doctor's diagnosis. It is a very challenging problem for the EHR owner fully controls over own EHR data as well as preserves the privacy of himself. In this paper, we propose a new privacy-preserving access control (PPAC) scheme for EHR. To achieve fine-grained access control of the EHR data, we utilize the attribute-based signcryption (ABSC) mechanism to signcrypt data based on the access policy for the linear secret sharing schemes. Employing the cuckoo filter to hide the access policy, it could protect the EHR owner's privacy information. In addition, the security analysis shows that the proposed scheme is provably secure under the decisional bilinear Diffie-Hellman exponent assumption and the computational Diffie-Hellman exponent assumption in the standard model. Furthermore, the performance analysis indicates that the proposed scheme achieves low costs of communication and computation compared with the related schemes, meanwhile preserves the EHR owner's privacy. Therefore, the proposed scheme is better suited to EHR system.

Keywords: electronic health records; privacy preserving; access control; attribute-based signcryption; cuckoo filter

1. Introduction

With the speedy growth of new-generation information techniques like the cloud computing and Internet of Things, and the uninterrupted improvement of living standards of people, the concept of smart city has also got more attention. In particular, the electronic health records (EHR) system has been widely applied in smart city since its appearance, and it has gradually been developed and improved [1,2]. However, in face of the tremendous EHR data, a third-party platform is needed to store and manage these data. Cloud computing provides inexpensive distributed computing capabilities through the Internet, which has the characteristics of ultra-large-scale and low-cost. Hence, managing and storing the EHR data in cloud servers has become an inevitable trend. In EHR system, EHR owners generally upload and view their personal information, medical records and medication records from cloud servers. Storing the EHR data in cloud servers which improves the quality of personal medical health management while saving resources and reducing hospital expenses. Only authorized EHR users (such as doctors or nurses) are able to log in the cloud servers and access data.

Although there are many significant advantages when using cloud servers to manage the EHR data, it also brings some concerns, such as the security and privacy of the sensitive data [3–5]. If a malicious and unauthorized adversary breaks the EHR system and conducts a series of malicious actions, including leaking patient's identity information and maliciously tampering with medical

records, it will not only result in disclosure of patient personal privacy, but also lead to misdiagnosis by the doctors and brings serious consequences. Hence, it is necessary to put forward the access control requirements to legitimate users who can access the EHR data. Attribute-based encryption (ABE) is employed to supply fined-grained access control of the EHR data. The EHR owner defines the access policy to determine who is capable to obtain the EHR data and uploads them to the cloud servers after encrypting it using the access policy. The ciphertext could be decrypted simply if the attributes of the EHR user meet the access policy that is defined by the EHR owner. Such as, the encryption access policy is "Alice" $\vee$ "XXX Hospital $\wedge$ Oncologist". So, the EHR owner named "Alice" or the EHR user who is the "oncologist" in "XXX hospital" has the right to access the EHR data.

Although ABE schemes [6–9] could provide secure access control for the EHR data in EHR system, they still suffer from a serious problem that the access policy may leak EHR owner's privacy. Here, the access policy will be send together with the ciphertext to EHR users in decryption phase, which may lead to the adversary gains owner's related sensitive information from the access policy. This is caused by the construction of access policy is related to the EHR owner's attributes. For instance, "Oncologist" is the sensitive information in the access policy for EHR owners. If anyone obtains this information, he might suspect that the EHR owner is suffering from oncology, which leads to the privacy leakage of the EHR owner. To achieve privacy-preserving for EHR system, some ABE schemes [10–17] were proposed.

However, all ABE schemes only support data encryption functionality and do not provide authentication capability. Attribute-based signcryption (ABSC) [18] mechanism emerges in integrating the fine-grained access control of data in attribute-based cryptography terminology and the efficient advantage of signcryption technology, which provides confidentiality, unforgeability and public verifiability simultaneously. Therefore, it is more appropriate to design a PPAC scheme for EHR system using the ABSC technology.

1.1. Our Contributions

In this paper, inspired by the ABSC mechanism and the cuckoo filter [19], a novel privacy-preserving access control (PPAC) scheme for EHR system is put forward. The major contributions are summed up as below:

- Based on the bilinear pairings, the ciphertext-policy attribute-based signcryption (CP-ABSC) scheme for EHR system is proposed. The proposed scheme ensures fined-grained access control of the EHR data, utilizes cuckoo filter to hide the access policy and preserves the privacy of EHR owners.
- The security analysis indicates that the proposed CP-ABSC scheme achieves the ciphertext indistinguishability and existential unforgeability in the standard model under the decisional bilinear Diffie-Hellman exponent (q-DBDHE) assumption and the computational Diffie-Hellman exponent (q-CDHE) assumption, respectively.
- The performance evaluation demonstrates that the proposed CP-ABSC scheme is more efficient than the related existing schemes [20–23] in terms of communication overheads and computation costs, and is right suitable for EHR system.

1.2. Organization

This paper is organized as below. The related work is described in Section 2. The preliminaries are reviewed in Section 3. The system model and security model are described in Section 4. The proposed PPAC scheme is given in Section 5. Sections 6 and 7 present the security proof and performance analysis, respectively. Finally, this paper is concluded in Section 8.

2. Related Works

Access control is a basic security service in modern computing systems. The access control management ensures that only authorized users are given access to certain resources, which is an effective method to protect data privacy. It is characterized by different access permissions and level of views, and usually constructed according to hierarchical scheme. In particular, Akl and Taylor [24] first proposed the use of cryptography to implement access control in hierarchical structures in 1983. Crampton et al. [25] introduced a novel cryptographic scheme to execute the enforcement of information flow policies. The advantage of this scheme is that no public information is needed to derive the decryption keys. Moreover, when performing a given policy, this tree-based scheme requires fewer keys compared to existing chain-based approaches. Castiglione et al. [26] not only explored the relationship between all the security concepts in the hierarchical key assignment scheme (HKAS), but also proposed a general architecture for HKAS, which provides security for strong key recovery and gives any HKAS that guarantees security for key recovery. According to the security and privacy of outsourced data, a large number of users must create, share, update and delete it dynamically, Castiglione [27] provided some new results on Akl and Taylor's scheme [24], for flexible and fine-grained access control to support dynamic updates in cloud environments. Alderman [28] designed a space-efficient KAS based on a binary tree, which eliminates public information as well as imposes logarithmic bounds on the number of derivatives required. This scheme performs better than the existing scheme, reduces the storage requirement of user equipment and logarithmically limits the derivation cost.

In 2005, the idea of ABE was proposed by Sahai and Waters [29], which is a one-to-many encryption mechanism. In this scheme, the users encrypt plaintext message based on the certain access control policy and adopt the attributes to identify user's identities. Afterwards, ABE is divided into ciphertext-policy ABE (CP-ABE) and key-policy ABE (KP-ABE) depending on whether the access structure is associated with the ciphertexts or the sceret keys, respectively. In 2006, the KP-ABE scheme was proposed by Goyal et al. [30], which supports delegation of private keys and provides flexible access policies that enable fine-grained access control. In 2007, Bethencourt et al. [31] constructed the CP-ABE scheme. Even though the storage server is not trusted, this scheme can keep the data confidentiality. In addition, this method could resist collusion attacks. Based on linear secret sharing schemes, Waters [32] firstly put forward a fully expressed CP-ABE scheme in the standard model. The sender of message can formulate an access policy according to its own attributes and define different access policies for different messages in this scheme. The CP-ABE schemes are more appropriate for access control applications, although both KP-ABE and CP-ABE schemes are able to utilize access policies to encrypt message and achieve access control of data. With the development of research, lots of ABE schemes [6–17,33–35] have been presented.

For guaranteeing the EHR data's confidentiality in data storage and transmission process, EHR owners must consider the access control of the EHR data with the aim at ensuring merely authorized users can obtain the important information. In 2009, Ibraimi et al. [6] present a novel CP-ABE scheme for safely managing and sharing the EHR data from an un-trusted web server, which is used to force organizational/patient access control policies and protect the data. In 2010, based on cryptographic constructions, Sun et al. [7] proposed a secure EHR system, which combining the mechanisms for revocation and fine-grained access control, and gives support for patient data secure sharing. In 2011, Akinyele et al. [8] designed a self-protecting EHR scheme employing ABE, the main purpose is that the access control policy may be assigned to each encrypted project. In 2013, Li et al. [9] gave a new secure EHR data sharing scheme in cloud computing, which simplified key management for users by using the multi-authorized ABE technique.

Owing to the sensitivity of health relevant data, offering privacy-preserving of EHR owners and access control of the EHR data is the main challenge in nowadays EHR system. Based on public key encryption with keyword search, Narayan et al. [10] proposed an ABE scheme to provide privacy preservation for EHR management system. An attribute-oriented authentication scheme was proposed

by Liang et al. [12], which is able to assist an EHR user to establish social relationships and share health information with other trusted users. Lu et al. [13] introduced the user-centric privacy access control scheme and allowed a medical user to determine who may take part in computing to give assistance to the EHR data processing. Liu et al. [14] proposed the online/offline ABE. EHR owners performed most of the encryption calculations during the offline encryption phase. When the access policy and the EHR data were known during the online encryption phase, EHR owners can quickly integrate information to generate the final ciphertext. Zhou et al. [15] presented two anonymous ABE schemes, which can achieve anonymity for personal EHR. On the basis of ABE, a PPAC scheme in mobile healthcare social networks was proposed by Jiang et al. [16]. In this scheme, they adopt bloom filter to hide attributes and efficiently query attributes before decryption. Yang et al. [17] constructed a new attribute bloom filter for the privacy-preserving CP-ABE scheme.

Combining the encryption and digital signature functions in a single step, Zheng [36] firstly proposed the concept of signcryption. And its advantages include that the communication overhead is much smaller than the steps of encryption and signature and it can achieve both confidentiality and authenticity. Combining the idea of ABE and signcryption, attribute-based signcryption (ABSC) has been put forward [18,20–23,37–44]. In 2010, Gagné et al. [18] proposed the ABSC scheme using the threshold access policy. In which, the users have to determine their access structure in advance in setup phase. In 2011, Wang et al. [20] put forward a ciphertext-policy and claim-predicate ABSC scheme based on bilinear pairings. Its efficiency is much higher than that of the combination of the cipertext policy attribute-based signature (CP-ABS) and CP-ABE. In 2012, the dynamic CP-ABSC scheme was proposed by Emura et al. [21], which allows the signature access structure updating without re-sending the user's signature key. This is the public verifiability, which permits any intermediary to check the validity of ciphertext before sending it to recipient. In 2013, a novel and security fuzzy attribute-based signcryption scheme was constructed by Hu et al. [22], which enables data encryption, access control, and digital signature for patient medical information in the body area networking. Afterward, based on the bilinear pairings on elliptic curves, Guo et al. [38] realized the concept of ring signcryption in the attribute-based encryption frame and present attributed-based ring singcryption scheme. Wang et al. [39] point that the ABSC scheme [18] is not secure under certain forgery. Han et al. [40] used the inner-product encryption and constructed a threshold ABSC scheme with constant-size ciphertext. In 2014, Wei et al. [41] designed a traceable ABSC scheme. This scheme's advantage is that the authority could breach anonymity of the signcryption while it is required to trace messages. In 2016, in the light of expressive LSSS access structure, Rao et al. [43] presented an efficient and constant-size ciphertext KP-ABSC scheme. To solve the problem of secure sharing fine-grained access control of the personal health records (PHR) data , Liu et al. [44] proposed a CP-ABSC scheme. Unfortunately, Rao et al. [23] pointed out the problems in scheme [44] and proposed a secure CP-ABSC scheme for the EHR data sharing in cloud.

In summary, the above mentioned ABSC schemes provide the confidentiality and unforegability of the EHR data. However, these schemes cannot specifically solve the problem about the privacy leakage of EHR owners in EHR system. Moreover, the access policies are still in the form of plaintext in these schemes. To a certain extent, the disclosure of the personal privacy information is still a challenging problem in the fine-grained data access control for EHR system.

Besides, now there are many cloud servers supporting two-factor authentication technology. Based on the analysis of the shortcomings of existing two-factor authentication schemes for privacy preserving, Wang et al. [45] proposed an efficient and provably secure two-factor authentication scheme in the random oracle model, which can achieve higher security and privacy without increasing communication or computing costs. In the following study, Wang et al. [46] proposed a two-factor authentication scheme in the random oracle model, which achieves security guarantees beyond the conventional optimal security bound. If an attribute-based authenticated key agreement scheme is constructed on the basis of signcryption technology, it can also provide good security and efficiency in PPAC scheme. In our research, we prefer to design a PPAC solution for EHR system under the

standard model. Therefore, in this paper, using the CP-ABSC scheme, we will present the PPAC scheme for the practical and secure EHR system, which prevent the leakage of EHR owner's personal privacy information from the access policy and may achieve fine-grained access control of EHR data.

3. Preliminaries

3.1. Bilinear Pairings

Let $\mathbb{G}$, $\mathbb{G}_T$ be two multiplicative cyclic groups of prime order p and g be the generator of $\mathbb{G}$. The bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ satisfies the following three properties:

1. Bilinearity: For all $u, v \in \mathbb{G}$ and $a, b \in \mathbb{Z}_p$, where $e(u^a, v^b) = e(u, v)^{ab}$.
2. Non-degeneracy: $e(g, g) \neq 1$.
3. Computability: For all $u, v \in \mathbb{G}$, there exists an efficient algorithm to compute $e(u, v)$ for all $u, v \in \mathbb{G}$.

3.2. Access Structures

Suppose $P = \{P_1, P_2, \cdots, P_n\}$ is a set of parties. There exists a collection $W \subseteq 2^P$, which is monotone if and only if for any set B and C, if $B \in W$ and $B \subseteq C$ then $C \in W$. An access structure is a collection W of non-empty subsets of $\{P_1, P_2, \cdots, P_n\}$, i.e., $W \subseteq 2^P \backslash \{\emptyset\}$. The sets in W are named as the authorized sets, otherwise which are named as the unauthorized sets.

3.3. Linear Secret Sharing Schemes

A secret sharing scheme Π for access structure W is called the linear secret sharing scheme (LSSS) over a set of parties P in $\mathbb{Z}_p$ if

1. The shares for each party form a vector over $\mathbb{Z}_p$.
2. There exists a share-generating matrix M with l rows and n columns for Π. For all $i = [1, l]$, $\rho(i)$ maps the i'th row of M to every authorized role attribute, where the function ρ is a function from $\{1, 2, \cdots, l\}$ to P. We find a column vector $\vec{v} = (\sigma, r_2, \cdots, r_n)$ be a sharing vector, where $r_2, \cdots, r_n \in \mathbb{Z}_p$ are random values and $\sigma \in \mathbb{Z}_p$ is the secret value to be shared. $M\vec{v}$ is the vector of l shares of σ on Π. Each $\lambda_i = (M\vec{v})_i$ is distributed as secret share value to each attribute $\rho(i)$.

An LSSS to be represented by an access structure $W = (M, \rho)$ is shown in Figure 1. Each LSSS has the linear reconstruction property, defined as follows: Let W be the access structure and Π be the LSSS. For any authorized set, i.e., $S \in W$, let $I = \{i : \rho(i) \in S\} \subset \{1, 2, \cdots, l\}$. According to Π, if $\{\lambda_i\}_{i \in I}$ are valid shares for the secret σ, here exists constants $\{w_i \in \mathbb{Z}_p\}_{i \in I}$ such that $\sum_{i \in I} w_i \lambda_i = \sigma$. Let M_i denote i'th row of M, then $\sum_{i \in I} w_i M_i = (1, 0, \cdots, 0)$. It is worth noting that the constants $\{w_i\}$ can be obtained in time polynomial in scale of the share-generation matrix M.

$$(M, \rho) \quad \overset{M}{\underset{l}{\begin{bmatrix} m_{1,1} & m_{1,2} & \cdots & \cdots & m_{1,n} \\ m_{2,1} & m_{2,2} & \cdots & \cdots & m_{2,n} \\ \vdots & \vdots & \ddots & & \vdots \\ \vdots & \vdots & & \ddots & \vdots \\ m_{l,1} & m_{l,2} & \cdots & \cdots & m_{l,n} \end{bmatrix}}}^{n} \quad \rightarrow \quad \overset{\rho}{\begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ \vdots \\ a_l \end{bmatrix}}$$

Figure 1. The LSSS access policy.

3.4. Cuckoo Filter

The data structure called cuckoo filter [19] is the extended version of bloom filter, which supports adding and removing items dynamically while having lower space overhead, shorter search time and better performance than bloom filter [47]. It also solves the problem of false positive in bloom filter. As a method for testing set membership, cuckoo filter uses cuckoo hashing technique [48] to solve the problem of false positive in bloom filter and check whether an element exists in a set.

Figure 2a shows the basic cuckoo hashing table that includes a series of buckets, and each bucket contains 4 entries. There are two candidate buckets in every item x, which are calculated from the formula and $h_1(x)$ and $h_2(x)$. The process of inserting a new element into the hash table is displayed as Figure 2b. In Figure 2, the hash table has 8 buckets. When adding a new element into the candidate bucket 1 or 5, if either of the two candidate buckets is empty, we will insert it into the other free bucket.If both buckets have no space the element selects any candidate bucket (such as "1") and removes the existing element, then this moved element need to re-insert into itself alternative position as shown in Figure 2b. In this case, it will trigger the item "c" that removes from bucket 3 into bucket 6 when removing "a". We will repeat this operation until we find an empty bucket and the maximum number of times is reached. When no empty bucket is obtained, the cuckoo hashing table will be regard as that it is too filled to insert.

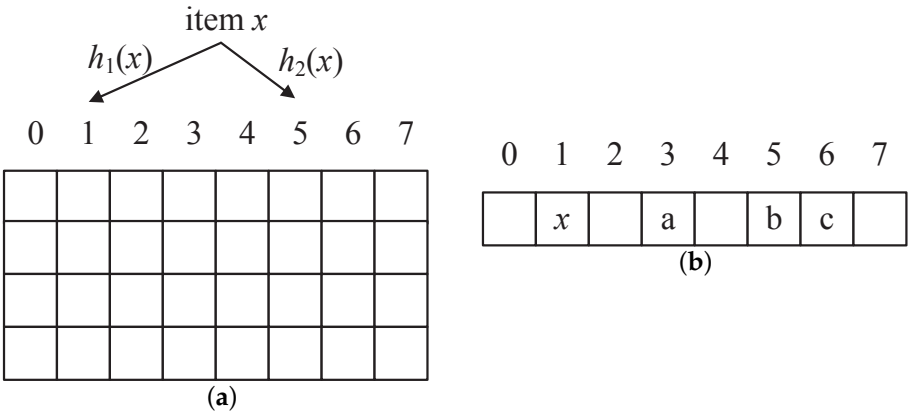

Figure 2. Cuckoo hashing table. (**a**) the basic cuckoo hashing table; (**b**) inserting a new element.

A cuckoo filter algorithm has mainly three functions: the insert function that stores items into the filter, the lookup function that checks whether an item exists in the filter and the delete function that removes the previously inserted items. For each item x, cuckoo filter stores a fingerprint and calculates two candidate buckets i_1 and i_2 by the following formulas:

$$i_1 = H_4(x) \tag{1}$$

$$i_2 = i_1 \oplus H_4\left(fingerprint(x)\right) \tag{2}$$

where H_4 is a one-way hash function.

We only adopt the insert and lookup functions of cuckoo filter in our paper. Algorithm 1 and Algorithm 2 illustrate the insert operation and lookup operation, respectively.

In Algorithm 1, cuckoo filter adds new items dynamically through storing fingerprints f of every item x. In Algorithm 2, we can easily check whether an item y belong to cuckoo filter.

Algorithm 1 Insert (x)

$f = \text{fingerprint}(x)$;
$i_1 = H_4(x)$;
$i_2 = i_1 \oplus H_4(x)$;
If bucket $[i_1]$ or bucket $[i_2]$ has an empty entry **then**
add f to that bucket;
return Done;
i= randomly pick i_1 or i_2;
For $n = 0$; $n <$ MaxMumKicks; $n++$ **do**
randomly select an entry e from bucket $[i]$;
swap f and fingerprint stored in entry e;
$i = i \oplus H_4(f)$;
If bucket $[i]$ has an empty entry **then**
add f to bucket $[i]$;
return done;
return False.

Algorithm 2 Lookup (y)

$f = \text{fingerprint}(y)$;
$i_1 = H_4(x)$;
$i_2 = i_1 \oplus H_4(x)$;
If bucket $[i_1]$ or bucket $[i_2]$ has f **then**
return True;
else
return False;
End If.

3.5. Complexity Assumptions

Decisional q-Bilinear Diffie-Hellman Exponent (q-DBDHE) Problem: Given the tuple $y_{a,\sigma} = (g, g^{\sigma}, g^{a}, g^{a^2}, \cdots, g^{a^q}, g^{a^{q+2}}, \cdots, g^{a^{2q}})$ in group $\mathbb{G}$ and $a, \sigma \in \mathbb{Z}_p$ are chosen at randomly, the task of q-DBDHE problem is to distinguish $e(g^{a^{q+1}}, g^{\sigma}) \in \mathbb{G}_T$ from a random element $R \in \mathbb{G}_T$.

The advantage of $\mathcal{A}$ in solving the q-DBDHE problem is defined as

$$Adv_{\mathcal{A}}^{q-\text{DBDHE}} = \Pr\left[1 \leftarrow \mathcal{A}(y_{a,\sigma}, T) | T = e(g,g)^{a^{q+1}\sigma}\right] - \Pr\left[1 \leftarrow \mathcal{A}(y_{a,\sigma}, T) | T = R\right] \geq \varepsilon.$$

q-DBDHE Assumption: It says that there is no known polynomial-time algorithm $\mathcal{A}$ to solve the q-DBDHE problem with advantage at least ε.

Computational q-Diffie-Hellman Exponent (q-CDHE) Problem: Given the tuple $y_a = (g, g^{a}, g^{a^2}, \cdots, g^{a^q}, g^{a^{q+2}}, \cdots, g^{a^{2q}})$ in group $\mathbb{G}$ and $a \in \mathbb{Z}_p$ is chosen at randomly, the task of q-CDHE problem is to compute $g^{a^{q+1}}$.

The advantage of in solving the q-CDHE problem is defined as

$$Adv_{\mathcal{A}}^{q-CDHE} = \Pr\left[g^{a^{q+1}} \leftarrow \mathcal{A}(y_a)\right] \geq \varepsilon.$$

q-CDHE Assumption: It says that there is no known polynomial-time algorithm $\mathcal{A}$ to solve the q-CDHE problem with advantage at least ε.

4. Model

In this section, we first give the typical structure of the EHR system model and the specific working stages of the proposed PPAC scheme for the EHR system model. Then, we define a CP-ABSC scheme and its security model, which is the basic method to implement the proposed PPAC scheme.

4.1. System Model

A typical structure of EHR system model is demonstrated in Figure 3.

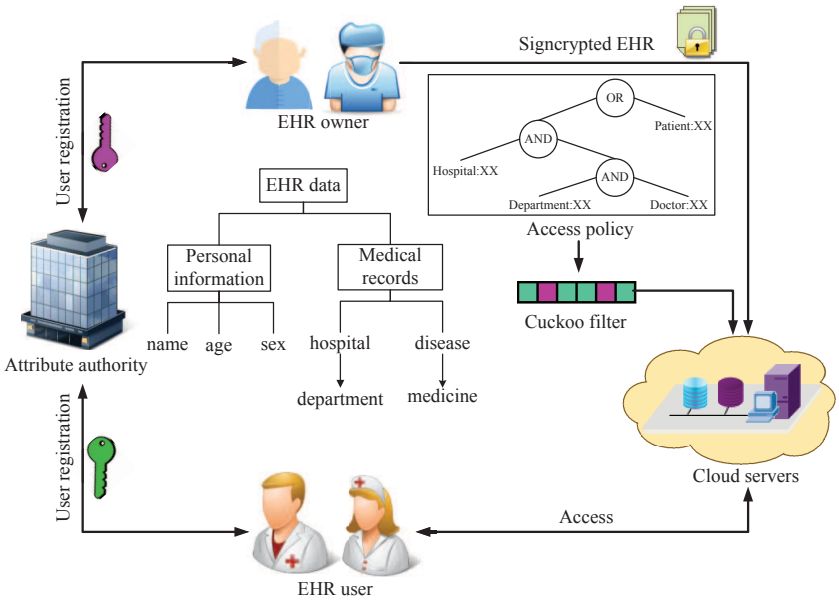

Figure 3. A framework of the EHR system.

EHR system comprises four entities: Attribute authority (AA), EHR owner, EHR user and Cloud servers.

- AA is a trusted party that is responsible for generating and distributing public parameters and private keys for the users, selects attributes from the attribute space and assigns to the users with different rights.
- EHR owner is the EHR data provider (such as a patient) who formulates the access policy, signcrypts his/her own EHR data and uploads the ciphertext to cloud servers.
- EHR user is the EHR data receiver (such as a doctor or nurse) who can download the cipgertext from cloud servers and unsigncrypt it.
- Cloud servers are in charge of storing ciphertext data that sent by the EHR owner and granting access rights to EHR users.

On the basis of the above EHR system model, our paper designs a new PPAC scheme for the EHR system, which includes the following four phases.

- **System initialization phase**: AA generates the master key and public systems parameters for EHR system, and then publishes the system parameters to all users (EHR owners and EHR users).
- **Users registration phase**: The users submit a registration application to AA. AA verifies the legitimacy of the identity of the user according to the attributes owned by itself and distributes corresponding private key to the user.
- **EHR signcrypt phase**: An EHR owner signcrypts the EHR data (such as personal information and medical records) under the access policy, hides the access policy by the cuckoo filter and uploads the ciphertext to cloud servers for data sharing.
- **EHR access phase**: An EHR user submits the data access request to the cloud servers, who can download ciphertext from cloud servers and unsigncrypt data to obtain original messages if and only if the attribute set of EHR user that satisfies access policy.

4.2. Security Model

The CP-ABSC scheme is composed of the following five algorithms [23,29]:

Setup: Given a security parameter k, system attribute set S and message universe $\mathcal{M}$, the algorithm outputs the master key MSK and system public parameters PK.

sExtract: Given PK, MSK and the signing attribute set $A_s \subseteq S$, the algorithm outputs the corresponding signing private key SK_{A_s}.

dExtract: Given PK, MSK and the decryption attribute set $A_d \subseteq S$, the algorithm outputs the corresponding decryption private key SK_{A_d}.

Signcrypt: Given PK, the message $m \in \mathcal{M}$, the signing private key SK_{A_s} for A_s, the encryption access structure $W_e = (M_e, \rho_e)$, signing access structure $W_s = (M_s, \rho_s)$, where $A_s \in W_s$, and the cuckoo filter, the algorithm outputs the ciphertext CT.

Unsigncrypt: Given PK, the ciphertext CT and the decryption private key SK_{A_d} for A_d, the algorithm firstly queries the corresponding attributes values by cuckoo filter and reconstructs the access structure $W'_e = (M_e, \rho'_e)$, and outputs message m if $A_d \in W'_e$. Otherwise, the algorithm returns $\perp$.

According to [23,32], the security of CP-ABSC needs to satisfy confidentiality and unforgeability.

The confidentiality (indistinguishability against adaptive chosen ciphertext attack (IND-CCA2)) for CP-ABSC is captured by an interactive game between the adversary $\mathcal{A}$ and the challenger $\mathcal{C}$ as follows.

Initialization: The adversary $\mathcal{A}$ chooses an encryption access structure W_e^* for the encryption attribute set A_d, which is applied to calculate the challenge ciphertext and provides it to the challenger $\mathcal{C}$.

Setup: $\mathcal{C}$ executes the **Setup** algorithm. $\mathcal{C}$ keeps the master key MSK secretly and returns the public parameters PK to $\mathcal{A}$.

Phase 1: $\mathcal{A}$ adaptively issues the following polynomial bounded queries.

- **sExtract queries**: Given a query on the signing attribute set A_s, $\mathcal{C}$ executes the **sExtract** algorithm and returns the corresponding private key SK_{A_s} to $\mathcal{A}$.
- **dExtract queries**: Given a query on the decryption attribute set $A_d \notin W_e^*$, $\mathcal{C}$ executes the **dExtract** algorithm and returns the corresponding private key SK_{A_d} to $\mathcal{A}$.
- **Signcrypt queries**: Given a query on the message $m \in \mathcal{M}$, the decryption attribute set A_d, the signing attribute set A_s, the encryption access structure W_e, the signing access structure W_s and cuckoo filter, $\mathcal{C}$ executes the **sExtract** algorithm and obtains the signing private key SK_{A_s}. Then $\mathcal{C}$ execute the **Signcrypt** algorithm to generate the ciphertext CT and returns to $\mathcal{A}$.
- **Unsigncrypt queries**: Given a query on the ciphertext CT, the decryption attribute set A_d and the signing attribute set A_s, $\mathcal{C}$ firstly queries the corresponding attributes of EHR users that are in cuckoo filter or not and reconstructs the access structure $W'_e = (M_e, \rho'_e)$. $\mathcal{C}$ executes the **dExtract** algorithm and obtains the decryption private key SK_{A_d}. And $\mathcal{C}$ executes the **Unsigncrypt** algorithm to obtain the message m and returns to $\mathcal{A}$.

Challenge: After completing the Phase 1, $\mathcal{A}$ outputs two equal length messages m_0^*, m_1^* and the signing access structure W_s^*. When the signing attribute set $A_s^* \in W_s^*$, $\mathcal{C}$ gets $SK_{A_d^*}$ by running the **dExtract** algorithm. $\mathcal{C}$ randomly chooses $\theta \in \{0, 1\}$ and executes the **Signcrypt** algorithm to generate the ciphertext CT^*. At last, $\mathcal{C}$ sends CT^* to $\mathcal{A}$ as its challenge ciphertext.

Phase 2: $\mathcal{A}$ adaptively issues the queries as in Phase 1 except the **dExtract queries** for any decryption attribute set $A_d \in W_e^*$ and the **Unsigncrypt queries** for the challenge ciphertext CT^* for any $A_d \in W_e^*$.

Guess: $\mathcal{A}$ outputs a guess bit $\theta \in \{0, 1\}$. If $\theta' = \theta$, $\mathcal{A}$ wins the above game.

The advantage of $\mathcal{A}$ that wins the above game is defined to be $Adv = |\Pr[\theta' = \theta] - \frac{1}{2}|$.

Definition 1(Confidentiality). A CP-ABSC scheme is IND-CCA2 security, if there is no polynomial-time adversary who wins the aforementioned game with the non-negligible advantage.

The unforgeability (existential unforgeability against adaptive chosen message attack (EUF-CMA)) for CP-ABSC is captured by an interactive game between the adversary $\mathcal{A}$ and the challenger $\mathcal{C}$ as follows.

Initialization: The adversary $\mathcal{A}$ provides the challenge signing access structure W_s^* to the challenger $\mathcal{C}$.

Setup: $\mathcal{C}$ executes the **Setup** algorithm. Then $\mathcal{C}$ keeps the master key MSK secretly and returns the public parameters PK to $\mathcal{A}$.

Query phase: $\mathcal{A}$ performs a polynomial bounded number of queries adaptively.

- **sExtract queries**: Give a query on the signing attributes set $A_s \notin W_s^*$, $\mathcal{C}$ executes the **sExtract** algorithm and returns the corresponding private key SK_{A_s} to $\mathcal{A}$.
- **dExtract queries**: Give a query on the decryption attributes set A_d, $\mathcal{C}$ executes the **sExtract** algorithm and returns the corresponding private key SK_{A_d} to $\mathcal{A}$.
- **Signcrypt queries**: Same as the **Signcrypt queries** in the confidentiality game.
- **Unsigncrypt queries**: Same as the **Unsigncrypt queries** in the confidentiality game.

Forgery: $\mathcal{A}$ outputs the forgery ciphertext CT^* on (m^*, W_s^*, W_e^*).

$\mathcal{A}$ wins above game if CT^* is valid and $\mathcal{A}$ never makes the **Signcrypt queries** on (m^*, W_s^*, W_e^*).

The advantage of $\mathcal{A}$ that wins the above game is defined as the probability that it wins the unforgeability game.

Definition 2(Unforgeability). A CP-ABSC scheme is EUF-CMA security, if there is no polynomial-time adversary who wins the aforementioned game with the non-negligible advantage.

5. The Proposed Scheme

The construction of PPAC scheme for EHR system is based on the CP-ABSC scheme and the concrete CP-ABSC scheme is given based on the bilinear pairing, supporting the linear secret sharing schemes. Employing the cuckoo filter to hide the access policy, it could protect the EHR owner's privacy information. The proposed scheme meets the requirements of PPAC in this section, by using CP-ABSC mechanism to signcrypt plaintext messages can satisfy the confidentiality and unforegability of the EHR data. At the same time, the use of cuckoo filter achieves the purpose of privacy preserving. Specifically, our proposed CP-ABSC scheme includes four phases: system initialization, user registration phase, EHR signcrypt phase and EHR access phase. The detail steps are as follows.

5.1. System Initialization

AA generates the master key MSK and public parameters PK for EHR system through executing the **Setup** algorithm.

- **Setup**: Given the security parameter k, message universe $\mathcal{M} : \{0,1\}^*$ and attribute set S that includes the EHR owner's attributes and EHR user's attributes. AA picks three collision resistant cryptographic hash functions: $H_1 : \{0,1\}^* \to \{0,1\}^l$, $H_2 : \mathbb{G} \to \mathbb{Z}_p^*$, $H_3 : \{0,1\} \to \mathbb{Z}_p^*$. Besides, AA chooses a one-way hash function $H_4 : \{0,1\} \to \mathbb{Z}_p^*$, which will be used to hash all $\rho(i)$ for $i \in \{1,2,\cdots,l\}$ in the access policy $W = (M, \rho)$ associated with the EHR owners' attributes. Then, AA randomly chooses $a, \alpha \in \mathbb{Z}_p^*$, $\delta_1, \delta_2, y_0, y_1, \cdots, y_l \in \mathbb{G}$ and sets $Y = e(g,g)^\alpha$. For each attribute $x \in S$, AA samples $h_x \in \mathbb{G}$.

The system parameters are $PK = \{\mathcal{M}, S, H_1, H_2, H_3, H_4, g^a, \delta_1, \delta_2, y_0, \{y_i\}_{i\in[1,l]}, Y, \{h_x\}_{x\in S}\}$ and the master key is $MSK = \{g^\alpha\}$.

5.2. User Registration Phase

According to the attributes of the EHR owner and the EHR user, AA generates the corresponding private keys through executing the **sExtract** and **dExtract** algorithms.

- **sExtract**: Given PK, MSK and the signing attribute set $A_s \subseteq S$, AA randomly selects $r_s \in \mathbb{Z}_p^*$ and outputs the EHR owner's signing private key $SK_{A_s} : K_s = g^{\alpha} g^{ar_s}$, $L_s = g^{r_s}$, $\{K_{s,x} = h_x^{r_s}\}_{x \in A_s}$.
- **dExtract**: Given PK, MSK and the decryption attribute set $A_d \subseteq S$, AA randomly picks $r_d \in \mathbb{Z}_p^*$ and outputs the EHR user's decryption private key $SK_{A_d} : K_d = g^{\alpha} g^{ar_d}$, $L_d = g^{r_d}$, $\{K_{d,x} = h_x^{r_d}\}_{x \in A_d}$.

5.3. EHR Signcrypt Phase

The EHR owner signcrypts his/her own EHR data and uses cuckoo filter to hide the access policy W associated with attributes through executing the **Signcrypt** algorithm.

- **Signcrypt**: Given the message $m \in \mathcal{M}$, the signing private key SK_{A_s}, and the encryption access policy $W_e = (M_e, \rho_e)$ and the signing access policy $W_s = (M_s, \rho_s)$ that are formulated by the EHR owner. The EHR owner performs the following steps.
  - The EHR owner selects a vector $\vec{v} = (\sigma, v_2, \cdots, v_n) \in \mathbb{Z}_p^*$ calculates $\lambda_i = \vec{v} \cdot M_i$ for $i = 1, 2, \cdots, l$, where M_i is the i'th row of matrix M. And the EHR owner randomly chooses $\varphi_i \in \mathbb{Z}_p$ and generators a vector $\vec{\varphi} = (-\varphi_1, -\varphi_2, \cdots, -\varphi_l)$ such that $\vec{\varphi} \cdot M_s = -\vec{1}_n$, that is $\sum_{i=1}^{l} \varphi_i \cdot M_{s,i} = -\vec{1}_n$, and $\varphi_i = 0$ for all i where $\rho_s(i) \notin A_s$, where $M_{s,i}$ is the i'th row of matrix M_s.
  - The EHR owner picks $\xi \in \mathbb{Z}_p^*$ and computes

$$C = mY^{\sigma}, C' = g^{\sigma}, \mu = H_2(C'), C'' = (\delta_1{}^{\mu}\delta_2)^{\sigma}, \{C_i = g^{a\lambda_i} h_{\rho_e(i)}^{-\sigma}\}_{i \in [1,l]},$$

$$S_1 = L_s = g^{r_s}, H_1(S_1, W_e, W_s) = (j_1, j_2, \cdots, j_l),$$

$$H_3(W_e, W_s, C, C', C'', \{C_i = g^{a\lambda_i} h_{\rho_e(i)}^{-\sigma}\}_{i \in [1,l]}) = \beta$$

$$S_2 = K_s \cdot \prod_{i \in [1,l]} (K_{s,\rho_s})^{\varphi_i} \cdot (y_0 \prod_{i \in [1,l]} y_i{}^{j_i})^{\sigma} \cdot (C'')^{\beta\xi}.$$

  - The EHR owner uses the cuckoo filter to hide the access policy $W_e = (M_e, \rho_e)$. In order to derive the alternative position of an item based on its fingerprint, it needs to utilize the partial-key cuckoo hashing [19]. That can ensure the EHR owner inserts new items to cuckoo filter dynamically. For each valid attribute $a_i \in S$, where the attribute $a_i = \rho_e(i)$ maps the i'th row of access matrix M, let item $x = a_i$. The EHR owner dynamically inserts a new item x into the cuckoo filter by using the insert operation as shown in Algorithm 1 and constructs the cuckoo filter data structure CF. Finally, the EHR owner uploads the ciphertext $CT = \{C, C', C'', \{C_i\}_{i \in [1,l]}, S_1, S_2, CF\}$ to the cloud server.

5.4. EHR Access Phase

In this phase, the EHR user downloads the ciphertext CT from the cloud servers, then gets message m through running the **Unsigncrypt** algorithm.

- **Unsigncrypt**: Given the ciphertext CT, the EHR user performs the following steps.
  - Suppose that S' is the attribute set of the EHR user. For every attribute $a_i' \in S'$, let an item $y = a_i'$. The EHR user first checks the attributes are in the access policy or not by using using the lookup operation of the cuckoo filter as shown in Algorithm 2. If the item y is in cuckoo filter, it means that the attribute a_i' exists in the access policy. Lastly, the EHR user generates the reconstructed attribute map $\rho_e'(i) = a_i'$ and obtains the access policy $W_e' = (M_e, \rho_e')$.

– The EHR user computes $\mu = H_2(C')$, $H_1(S_1, W'_e, W_s) = (j_1, j_2, \cdots, j_l)$, $\beta = H_3(W'_e, W_s, C, C', C'', \{C_i = g^{a\lambda_i} h_{\rho_e(i)}^{-\sigma}\}_{i\in[1,l]})$ and verifies

$$Y = \frac{e(S_2, g)}{e(g^a \cdot \prod\limits_{i\in[1,l]} h_{\rho_s(i)}^{\varphi_i}, S_1) \cdot e(y_0 \prod\limits_{i\in[1,l]} {y_i}^{j_i} \cdot (\delta_1{}^{\mu}\delta_2)^{\beta\xi}, C')} \tag{3}$$

– If it is invalid, returns $\perp$; Otherwise, when the decryption attribute set $A_d \in S'$ satisfies (M_e, ρ'_e), the EHR user finds the constants $\{\omega_i \in \mathbb{Z}_p^*\}_{i\in I}$ such that $\{\lambda_i\}$ are valid shares of secret value σ based on M_e, $\sum_{i\in I} \omega_i\lambda_i = \sigma$, where $I = \{i : \rho'_e(i) \in S'\}$.

The EHR user computes

$$Y^{\sigma} = \frac{e(C', K_d)}{\prod_{i\in I} \left(e(C_i, L_d) \cdot e(C', K_{d,\rho'_e})\right)^{\omega_i}} \tag{4}$$

and recovers the message m from $m = \frac{C}{Y^{\sigma}}$.

Correctness:

$$\begin{aligned} S_2 &= K_s \cdot \prod_{i\in[1,l]} (K_{s,\rho_s})^{\varphi_i} \cdot (y_0 \prod_{i\in[1,l]} {y_i}^{j_i})^{\sigma} \cdot (C'')^{\beta\xi} \\ &= g^{\alpha} g^{ar_s} \cdot \prod_{i\in[1,l]} (h_{\rho_s(i)}^{r_s})^{\varphi_i} \cdot (y_0 \prod_{i\in[1,l]} {y_i}^{j_i})^{\sigma} \cdot (C'')^{\beta\xi}, \end{aligned}$$

$$\begin{aligned} e(S_2, g) &= e(g^{\alpha} g^{ar_s}, g) \cdot e(\prod_{i\in[1,l]} (h_{\rho_s(i)}^{r_s})^{\varphi_i}, g) \cdot e((y_0 \prod_{i\in[1,l]} {y_i}^{j_i})^{\sigma}, g) \cdot e((\delta_1^{\mu}\delta_2)^{\sigma\beta\xi}, g) \\ &= e(g, g)^{\alpha} \cdot e(g^a, g^{r_s}) \cdot e(\prod_{i\in[1,l]} (h_{\rho_s(i)})^{\varphi_i}, g^{r_s}) \cdot e(y_0 \prod_{i\in[1,l]} y_i^{j_i}, g^{\sigma}) \cdot e((\delta_1^{\mu}\delta_2)^{\beta\xi}, g^{\sigma}) \\ &= Y \cdot e(g^a \prod_{i\in[1,l]} h_{\rho_s(i)}^{\varphi_i}, S_1) \cdot e(y_0 \prod_{i\in[1,l]} y_i^{j_i} \cdot (\delta_1^{\mu}\delta_2)^{\beta\xi}, C'), \end{aligned}$$

$$\begin{aligned} \frac{e(C', K_d)}{\prod_{i\in I} (e(C_i, L_d) \cdot e(C', K_{d,\rho'_e}))^{\omega_i}} &= \frac{e(g^{\sigma}, g^{\alpha} g^{ar_d})}{\prod\limits_{i\in I} (e(g^{a\lambda_i} h_{\rho'_e(i)}^{-\sigma}, g^{r_d}) \cdot e(g^{\sigma}, h_{\rho_{e'}(i)}^{r_d}))^{\omega_i}} \\ &= \frac{e(g,g)^{\alpha\sigma} \cdot e(g,g)^{a\sigma r_d}}{\prod\limits_{i\in I} e(g,g)^{ar_d\lambda_i\omega_i}} = e(g,g)^{\alpha\sigma} = Y^{\sigma}. \end{aligned}$$

6. Security Proof

6.1. Confidentiality

Theorem 1. *Assuming there is the adversary $\mathcal{A}$ who is capable of breaking the IND-CCA2 security of CP-ABSC scheme with a non-negligible probability ε, then we we can construct an algorithm $\mathcal{B}$ that solves the* q*-DBDHE problem with the probability at least $\varepsilon' = \varepsilon - \frac{q_{us}}{p}$, where q_{us} is the maximum number of the* ***Unsigncrypt queries*** *issued by $\mathcal{A}$.*

Proof. The algorithm $\mathcal{B}$ receives an instance $y_{a,\sigma} = (g, g^{\sigma}, g^a, g^{a^2}, \cdots, g^{a^q}, g^{a^{q+2}}, \cdots, g^{a^{2q}})$ of the q-DBDHE problem, where $g_i = g^{a^i}$, $a, \sigma \in \mathbb{Z}_p$ and g is a generator of $\mathbb{G}$. The goal of $\mathcal{B}$ is to decide whether $T = e(g,g)^{a^{q+1}\sigma}$ or $T = R$, where R is a random element in $\mathbb{G}_T$. If $T = e(g,g)^{a^{q+1}\sigma}$, $\mathcal{B}$ outputs 1; Otherwise outputs 0. Then $\mathcal{B}$ chooses three collision-resistant hash functions $H_1 : \{0,1\}^* \to \{0,1\}^l$, $H_2 : \mathbb{G} \to \mathbb{Z}_p^*$, $H_3 : \{0,1\} \to \mathbb{Z}_p^*$ and a one-way hash function $H_3 : \{0,1\} \to \mathbb{Z}_p^*$. The algorithm $\mathcal{B}$ simulates the challenger in IND-CCA2 security game and interacts with the adversary $\mathcal{A}$ as below. □

Initialization: $\mathcal{A}$ submits the message space $\mathcal{M}: \{0,1\}^*$ and the challenge encryption access structure $W_e^* = (M_e^*, \rho_e^*)$ to $\mathcal{B}$, where M_e^* is a matrix of $l^* \times n^*$ with the labeling function ρ_e^*. Let $\vec{M}_i^* = (M_{i,1}^*, M_{i,2}^*, \cdots, M_{i,n^*}^*)$ be the i'th row of M_e^*.

Setup: $\mathcal{B}$ chooses a random $\alpha' \in \mathbb{Z}_p^*$ and calculates $\alpha = \alpha' + a^{q+1}$, $Y = e(g,g)^{\alpha} = e(g^a, g^{a^q}) \cdot e(g,g)^{\alpha'}$. $\mathcal{B}$ randomly chooses $\varsigma \in \mathbb{Z}_p^*$, $\eta_0, \eta_1, \cdots, \eta_l \in \mathbb{Z}_p^*$ and sets $C'^* = g^{\sigma}$, $\mu^* = H_2(C'^*)$, $\delta_1 = g_q^{\frac{1}{\mu^*}}$, $\delta_2 = g^{\varsigma} g_q^{-1}$, $y_0 = g^{\eta_0}, y_1 = g^{\eta_1}, \cdots, y_l = g^{\eta_l}$.

Finally, for each attribute $x \in S$, let X denote the set of indices i such that $\rho_e^*(i) = x$. If $X \neq \emptyset$, $\mathcal{B}$ selects a random parameter $f_x \in \mathbb{Z}_p^*$ and defines $h_x = g^{f_x} \cdot g^{aM_{i,1}^*} \cdot g^{a^2 M_{i,2}^*} \cdots g^{a^{n^*} M_{i,n}^*}$. If $X = \emptyset$, then $h_x = g^{f_x}$.

$\mathcal{B}$ returns the public parameters $PK = \{S, \mathcal{M}, H_1, H_2, H_3, H_4, Y, \delta_1, \delta_2, y_0, \{y_i\}_{i\in[1,l]}, Y, \{h_x\}_{x\in S}\}$ to $\mathcal{A}$.

Phase 1: $\mathcal{A}$ adaptively makes a number of queries as follows.

- **sExtract queries**: When $\mathcal{A}$ issues a query on the signing attribute set A_s, $\mathcal{B}$ randomly chooses $\hat{r} \in \mathbb{Z}_p^*$, sets $r_s = \hat{r} - a^q$ and computes $L_s = g^{\hat{r}} g_q^{-1}$, $K_s = g^{\alpha'} g_1^{\hat{r}}$, $K_{s,x} = h_x^{\hat{r}} g_q^{-f_x}$ for any $x \in A_s$. Then $\mathcal{B}$ returns the signing private key $SK_{A_s} = \{L_s, K_s, \{K_{s,x}\}_{x\in A_s}\}$ to $\mathcal{A}$.

  Correctness:

$$L_s = g^{\hat{r}} g_q^{-1} = g^{\hat{r}} g^{-a^q} = g^{r_s},$$

$$K_s = g^{\alpha'} g_1^{\hat{r}} = g^{\alpha'} g_{q+1} g_1^{\hat{r}} g_{q+1}^{-1} = g^{\alpha' + a^{q+1}} g^{a\hat{r} - a^{q+1}} = g^{\alpha} g^{a r_s},$$

$$K_{s,x} = h_x^{\hat{r}} g_q^{-f_x} = h_x^{\hat{r}} (h_x)^{-a^q} = h_x^{r_s}.$$

- **dExtract queries**: When $\mathcal{A}$ issues a query on the decryption attributes set $A_d \notin W_e^*$, $\mathcal{B}$ randomly chooses a vector $\vec{\gamma} = (\gamma_1, \gamma_2, \cdots, \gamma_{n^*}) \in \mathbb{Z}_p^{n^*}$ where $\gamma_1 = -1$, $\vec{\gamma} \cdot M_{e,i}^* = 0$ for all i where $\rho_e^*(i) \in A_d$. $\mathcal{B}$ randomly selects $\hat{r} \in \mathbb{Z}_p^*$, implicitly defines $r_d = \hat{r} + \gamma_1 a^q + \gamma_2 a^{q-1} + \cdots + \gamma_{n^*} a^{q-n^*+1}$ and computes $L_d = g^{\hat{r}} \prod_{i=1}^{n^*} (g^{a^{q+1-i}})^{\gamma_i}$, $K_d = g^{\alpha'} g^{a\hat{r}} \prod_{i=2}^{n^*} (g^{a^{q+2-i}})^{\gamma_i}$ and $K_{d,x} = L_d^{f_x} \prod_{j=1}^{n^*} (g^{a^j \cdot \hat{r}} \prod_{\substack{o=1,\cdots,n^* \\ o\neq j}} (g^{a^{q+1+j-o}})^{\gamma_o})^{M_{i,j}^*}$ for any $x \in A_d$. For any $i \in [1, l_e^*]$, if there is no $\rho_e^*(i) = x$, then $\mathcal{B}$ simply sets $K_{d,x} = L_d^{f_x}$. Then $\mathcal{B}$ returns the decryption key $SK_{A_d} = \{L_d, K_d, \{K_{d,x}\}_{x\in A_d}\}$.

  Correctness:

$$L_d = g^{\hat{r}} \prod_{i=1}^{n^*} (g^{a^{q+1-i}})^{\gamma_i} = g^{r_d},$$

$$\begin{aligned} K_d &= g^{\alpha'} g^{a\hat{r}} \prod_{i=2}^{n^*} (g^{a^{q+2-i}})^{\gamma_i} = g^{\alpha'} g^{a^{q+1}} \cdot g^{a\hat{r}} \cdot g^{-a^{q+1}} \prod_{i=2}^{n^*} (g^{a^{q+2-i}})^{\gamma_i} = g^{\alpha} g^{a\hat{r}} \prod_{i=1}^{n^*} (g^{a^{q+2-i}})^{\gamma_i} \\ &= g^{\alpha} (g^a)^{\hat{r} + \sum_{i=1}^{n^*} (g^{a^{q+1-i}})^{\gamma_i}} = g^{\alpha} g^{a r_d}, \end{aligned}$$

$$\begin{aligned} K_{d,x} &= L_d^{f_x} \cdot \prod_{j=1}^{n^*} (g^{a^j \cdot \hat{r}} \prod_{\substack{o=1,\ldots,n^* \\ o\neq j}} (g^{a^{q+1+j-o}})^{\gamma_o})^{M_{i,j}^*} = g^{\hat{r} f_x} \prod_{i=1}^{n^*} (g^{a^{q+1-i}})^{\gamma_i f_x} \cdot \prod_{j=1}^{n^*} (g^{a^j})^{\hat{r} \cdot M_{i,j}^*} \\ &= (g^{f_x} \prod_{j=1}^{n^*} (g^{a^j})^{M_{i,j}^*})^{\hat{r}} \cdot \prod_{i=1}^{n^*} (g^{a^{q+1-i}})^{\gamma_i f_x} = h_x^{\hat{r}} \cdot \prod_{i=1}^{n^*} (h_x{}^{a^{q+1-i}})^{\gamma_i} = h_x^{r_d}. \end{aligned}$$

- **Signcrypt queries**: When $\mathcal{A}$ issues a query on (m, W_e, W_s, A_d, A_s) and the cuckoo filter, if signing attribute set $A_s \in W_s$, $\mathcal{B}$ runs the **sExtract queries** and gets the private key SK_{A_s}, then $\mathcal{B}$ executes

the **Signcrypt** algorithm, generates ciphertext $CT = \{C, C', C'', \{C_i\}_{i\in[1,l]}, S_1, S_2, CF\}$. Finally, $\mathcal{B}$ returns CT to $\mathcal{A}$.

- **Unsigncrypt queries**: When $\mathcal{A}$ issues a query on the ciphertext CT, $\mathcal{B}$ checks whether $C' = C'^*$. If $C' = C'^*$, $\mathcal{B}$ aborts. (Since $C' = g^{\sigma}$ is random, the probability is at most $1/p$). Otherwise, $\mathcal{B}$ first checks the corresponding attributes of EHR user are in cuckoo filter or not and reconstructs the encryption access policy $W_e'^* = (M_e^*, \rho_e'^*)$.

  - If $A_d \notin W_e'^*$, $\mathcal{B}$ generates the private key SK_{A_d} through executing the **dExtract queries** and returns the results of the **Unsigncrypt** algorithm to $\mathcal{A}$.
  - If $A_d \in W_e'^*$, $\mathcal{B}$ first checks the validity of ciphertext CT based on Equation (3). If it is not valid, then $\mathcal{B}$ outputs $\perp$; Otherwise computes $Y^{\sigma} = e(C''/C'^{\varsigma}, g_1)^{\left(\frac{\mu}{\mu^*}-1\right)^{-1}} \cdot e(C', g^{\alpha'})$. Finally, $\mathcal{B}$ returns the message $m = \frac{C}{Y^{\sigma}}$ to $\mathcal{A}$.

Correctness:

$$\begin{aligned} e\left(C''/C'^{\varsigma}, g_1\right)^{\left(\frac{\mu}{\mu^*}-1\right)^{-1}} \cdot e(C', g^{\alpha'}) &= e((\delta_1^{\mu}\delta_2)^{\sigma} \Big/ g^{\sigma\varsigma}, g_1)^{\left(\frac{\mu}{\mu^*}-1\right)^{-1}} \cdot e(C', g^{\alpha'}) \\ &= \left(\left(g_q^{\frac{1}{\mu^*}}\right)^{\mu^*} \cdot g^{\varsigma} g_q^{-1}\right)^{\sigma} \Big/ g^{\sigma\varsigma}, g_1)^{\left(\frac{\mu}{\mu^*}-1\right)^{-1}} \cdot e(C', g^{\alpha'}) \\ &= e(g_q^{\sigma}, g_1) \cdot e(C', g^{\alpha'}) \\ &= e(C', g^{a^{n+1}}) \cdot e(C', g^{\alpha'}) \\ &= e(C', g^{\alpha}). \end{aligned}$$

Since Equation (3) is valid , it has $e(g^{ar_d}, C') = \prod_{i\in I} e(g^{ar_d}, g)^{\lambda_i\omega_i}$. Therefore,

$$\begin{aligned} & e\left(C''/C'^{\varsigma}, g_1\right)^{\left(\frac{\mu}{\mu^*}-1\right)^{-1}} \cdot e(C', g^{\alpha'}) \\ &= e(C', g^{\alpha}) \cdot \frac{e(g^{ar_d}, C')}{\prod_{i\in I} e(g^{ar_d}, g)^{\lambda_i\omega_i}} = \frac{e(C', g^{\alpha} g^{ar_d})}{\prod_{i\in I} \left(e(g^{a\lambda_i} h_{\rho_e(i)}^{-\sigma}, g^{r_d}) \cdot e(g^{\sigma}, h_{\rho_e(i)}^{r_d})\right)^{\omega_i}} \\ &= \frac{e(C', K_d)}{\prod_{i\in I} \left(e(C_i, L_d) \cdot e(C', K_{d,x})\right)^{\omega_i}} = Y^{\sigma}. \end{aligned}$$

Challenge: $\mathcal{A}$ outputs two equal length messages $m_0^*, m_1^* \in \mathcal{M}$ and the signing access policy W_s^* to $\mathcal{B}$. $\mathcal{B}$ chooses $t_1' = 0, \tilde{r}, t_2', t_3', \cdots, t_{n^*}' \in \mathbb{Z}_p^*$ and sets $r_s = \tilde{r} - a^q$, $\vec{v} = (\sigma + t_1', \sigma a + t_2', \sigma a^2 + t_3', \cdots, \sigma a^{n^*-1} + t_{n^*}') = \sigma(1, a, a^2, \cdots, a^{n^*-1}) + (0, t_2', t_3', \cdots, t_{n^*}')$. Then $\mathcal{B}$ selects $\theta \in \{0, 1\}$ and outputs the challenge ciphertext $CT^* = (C^*, C'^*, C''^*, \{C_i{}^*\}_{i\in[1,l^*]}, S_1^*, S_2^*)$ as follows:

- $C^* = m_\theta T \cdot e(g^{\sigma}, g^{a'})$,
- $C'^* = g^{\sigma}$,
- $C''^* = (g^{\sigma})^{\varsigma}$, where $\mu^* = H_2(C'^*)$,
- $C_i{}^* = \left(\prod_{j=1}^{n^*} (g^a)^{M_{i,j}^* \cdot t_j'}\right) \cdot (g^{\sigma})^{-f_{\rho_e^*(i)}}$ for $i \in [1, l^*]$,
- $S_1^* = g^{\tilde{r}} g_q^{-1}$,
- $S_2^* = (g^{\alpha'} g^{a\tilde{r}}) \cdot (h_{\rho_s^*(i)}^{r_i^*} g_q^{-f_{\rho_s^*(i)}})^{\varphi_i^*} \cdot (g^s)^{\eta_0 + \sum_{i=1}^{l^*} j_i^* \eta_i + \varsigma\xi\beta^*}$, where $H_1(S_1^*, W_e^*, W_s^*) = (j_1^*, j_2^*, \cdots, j_l{}^*)$, $H_3(W_e^*, W_s^*, C, C'^*, C''^*, \{C_i{}^*\}_{i\in[1,l^*]}) = \beta^*$.

If $T = e(g^{\sigma}, g^{a^{q+1}})$, CT^* is a valid challenge ciphertext.

Correctness:

$$T \cdot e(g^{\sigma}, g^{\alpha'}) = e(g^{\sigma}, g^{a^{q+1}}) \cdot e(g^{\sigma}, g^{\alpha'}) = e(g,g)^{\sigma\alpha} = Y^{\sigma}.$$

$$C^* = m_{\theta} \cdot T \cdot e(g^{\sigma}, g^{a'}) = m_{\theta} \cdot Y^{\sigma}.$$

$$C''^* = (g^{\sigma})^{\varsigma} = (g^{\varsigma} g_q^{-1} g_q)^{\sigma} = ((g_q^{\frac{1}{\mu^*}})^{\mu^*} \cdot g^{\varsigma} g_q^{-1})^{\sigma} = (\delta_1^{\mu^*} \delta_2)^{\sigma}.$$

For $j = 1, 2, \cdots, n^*$, $\lambda_i = \vec{v} \cdot M_i{}^* = (\sigma(1, a, a^2, \cdots, a^{n^*-1}) + (0, t'_2, t'_3, \cdots, t'_{n^*})) \cdot M_i{}^* = a\sigma \sum_{j=1}^{n^*} a^{j-1} M^*_{i,j} + \sum_{j=2}^{n^*} t'_j M^*_{i,j}$,

$$\begin{aligned} C_i{}^* &= (\prod_{j=1}^{n^*} (g^a)^{M^*_{i,j} \cdot t'_j}) \cdot (g^{\sigma})^{-f_{\rho^*_e(i)}} \\ &= g^{\sigma \sum_{j=1}^{n^*} a^j M^*_{i,j}} \cdot (\prod_{j=1}^{n^*} (g^a)^{M^*_{i,j} \cdot t'_j}) \cdot (g^{\sigma})^{-f_{\rho^*_e(i)}} \cdot g^{-\sigma \sum_{j=1}^{n^*} a^j M^*_{i,j}} \\ &= ((g^a)^{\sigma \sum_{j=1}^{n^*} a^{j-1} M^*_{i,j}}) \cdot (\prod_{j=1}^{n^*} (g^a)^{M^*_{i,j} \cdot t'_j}) \cdot (g^{f_{\rho^*_e(i)}})^{-\sigma} \cdot g^{-\sigma \sum_{j=1}^{n^*} a^j M^*_{i,j}} \\ &= g^{a\lambda_i} \cdot (g^{f_{\rho^*_e(i)}} g^{\sum_{j=1}^{n^*} a^j M^*_{i,j}})^{-\sigma} = g^{a\lambda_i} h^{-\sigma}_{\rho^*_e(i)}. \end{aligned}$$

$S_1^* = g^{\tilde{r}} g_q^{-1} = g^{\tilde{r}} g^{-a^q} = g^{r_s} = L_s$.

$$\begin{aligned} S_2^* &= (g^{\alpha'} g^{a\tilde{r}}) \cdot (h^{r^*}_{\rho^*_s(i)} g_q^{-f_{\rho^*_s(i)}})^{\varphi_i^*} \cdot (g^s)^{\eta_0 + \sum_{i=1}^{l^*} j_i^* \eta_i + \varsigma\xi\beta^*} \\ &= g^{\alpha'} g^{a^{q+1}} g^{a\tilde{r}} g^{-a^{q+1}} \left(h^{\tilde{r}}_{\rho^*_s(i)} h^{-a^q}_{\rho^*_s(i)} \right)^{\varphi_i^*} \cdot (g^{\eta_0 + \sum_{i=1}^{l^*} j_i^* \eta_i})^{\sigma} \cdot \left((g^{\sigma})^{\varsigma}\right)^{\beta^*\xi} \\ &= g^{\alpha} g^{ar_s} \cdot h^{\varphi_i^*}_{\rho^*_s(i)} \cdot \left(y_0 \prod_{i=1}^{l^*} y_i^{j_i^*} \right)^{\sigma} \cdot (C''^*)^{\beta^*\xi} \\ &= K_s \cdot (K_{s,x})^{\varphi_i^*} \cdot \left(y_0 \prod_{i=1}^{l^*} y_i^{j_i^*} \right)^{\sigma} \cdot (C''^*)^{\beta^*\xi}. \end{aligned}$$

Phase 2: $\mathcal{A}$ performs a series of queries as **Phase 1** except the **dExtract queries** on any decryption attribute set $A_d \in W_e^*$ and the **Unsigncrypt queries** on the challenge ciphertext CT^* for any $A_d \in W_e^*$.

Guess: $\mathcal{A}$ outputs a guess bit $\theta' \in \{0,1\}$. If $\theta' = \theta$, $\mathcal{B}$ outputs 1 ($T = e(g,g)^{a^{q+1}\sigma}$); Otherwise $\mathcal{B}$ outputs 0 ($T = R$).

$\mathcal{B}$ can't successfully simulate with aborting the game when the ciphertext satisfies $C' = C'^*$ in the **Unsigncrypt queries**, the probability of this aborting event is at most $\frac{q_{us}}{p}$. If $\mathcal{B}$ doesn't abort and $T = e(g,g)^{a^{q+1}\sigma}$, the probability of the successful simulation for $\mathcal{B}$ is at least $\frac{1}{2} + \varepsilon - \frac{q_{us}}{p}$. If $T = R$, the probability of $\mathcal{A}$ does not get any information about m_{θ}^* is $\frac{1}{2}$. Therefore, the advantage of $\mathcal{B}$ can solve the q-DBDHE problem is at least $\varepsilon' = \Pr\left|\mathcal{B}(y, T = e(g,g)^{a^{q+1}\sigma}) = 0\right| - \Pr\left|\mathcal{B}\left(y, T = R\right) = 0\right| = \varepsilon - \frac{q_{us}}{p}$.

6.2. Unforgeability

Theorem 2. *Assuming there is the adversary $\mathcal{A}$ who is capable of breaking the EUF-CMA security of CP-ABSC scheme with the non-negligible probability ε, then we can construct an algorithm $\mathcal{B}$ that can solve* q*-CDHE problem with the probability $\varepsilon' = \varepsilon k(l+1)$, where k is the security parameter and l is the outputs length of hash function H_1.*

Proof. $\mathcal{B}$ receives an instance $y_a = (g, g^a, g^{a^2}, \cdots, g^{a^q}, g^{a^{q+2}}, \cdots, g^{a^{2q}})$ of the q-CDHE problem, where $a \in \mathbb{Z}_p$, g is a generator of $\mathbb{G}$ and $g_i = g^{a^i}$. The goal of the algorithm $\mathcal{B}$ is to calculate $g^{a^{q+1}}$. $\mathcal{B}$ chooses three collision-resistant hash functions $H_1 : \{0,1\}^* \to \{0,1\}^l$, $H_2 : \mathbb{G} \to \mathbb{Z}_p^*$, $H_3 : \{0,1\} \to \mathbb{Z}_p^*$ and a

one-way hash function $H_4 : \{0,1\} \to \mathbb{Z}_p^*$. $\mathcal{B}$ simulates the challenger in EUF-CMA security game and interacts with $\mathcal{A}$ as below. □

Initialization: $\mathcal{A}$ submits the challenge signing access policy $W_s^* = (M_s^*, \rho_s^*)$ to $\mathcal{B}$, where M_s^* is a matrix of $l^* \times n^*$ with the labeling function ρ_s^*. Let $\vec{M}_i^* = (M_{i,1}^*, M_{i,2}^*, \cdots, M_{i,n^*}^*)$ be the i'th row of M_s^*.

Setup: $\mathcal{B}$ randomly picks $\alpha' \in \mathbb{Z}_p^*$, $d, d' \in \mathbb{Z}_p^*$ and defines $\alpha = \alpha' + a^{q+1}$, $Y = e(g,g)^{\alpha} = e(g^a, g^{a^q}) \cdot e(g,g)^{\alpha'}$, $\delta_1 = g^d$, $\delta_2 = g^{d'}$. $\mathcal{B}$ randomly chooses $(z_0, z_1, \cdots, z_l) \in \mathbb{Z}_p^{l+1}$, $\eta = k$ and $\eta(l+1) < p$, where k is a security parameter. $\mathcal{B}$ also randomly selects $0 \le \pi \le l$ and $(b_0, b_1, \cdots, b_l) \in \mathbb{Z}_\eta^{l+1}$ sets $y_0 = g_q^{p-\eta\pi+b_0}$, $y_i = g_q^{b_i} g^{z_i}$ for all $i \in [1,l]$. For each vector $\vec{j} = (j_1, j_2, \cdots, j_l) \in \{0,1\}^l$, $\mathcal{B}$ defines two functions $F_1(\vec{j}) = p - \eta\pi + b_0 + \sum_{i=1}^{l} j_i b_i$ and $F_2(\vec{j}) = z_0 + \sum_{i=1}^{l} j_i z_i$, which means that $y_0 \prod_{i=1}^{l} y_i{}^{j_i} = g_q^{F_1(\vec{j})} g^{F_2(\vec{j})}$. $\mathcal{B}$ defines the function $F : \{0,1\}^l \to \{0,1\}$ by $F(\vec{j}) = \begin{cases} 0, & \text{if } b_0 + \sum_{i=1}^{l} j_i b_i = 0 \bmod \eta, \\ 1, & \text{otherwise.} \end{cases}$.

It can be seen that, if $F(\vec{j}) = 1$, then $F_1(\vec{j}) \neq 0 \bmod p$.

Finally, for each attribute $x \in S$, let X denote the set of indices i, such that $\rho_s^*(i) = x$. If $X \neq \emptyset$, $\mathcal{B}$ selects a random $f_x \in Z_p^*$ and defines $h_x = g^{f_x} \cdot g^{aM_{i,1}^*} \cdot g^{a^2 M_{i,2}^*} \cdots g^{a^{n^*} M_{i,n}^*}$. If $X = \emptyset$, then $h_x = g^{f_x}$.

$\mathcal{B}$ returns the public parameters $PK = \{S, \mathcal{M}, H_1, H_2, H_3, H_4, Y, \delta_1, \delta_2, y_0, \{y_i\}_{i\in[1,l]}, Y, \{h_x\}_{x\in S}\}$ to $\mathcal{A}$.

Query phase: $\mathcal{A}$ adaptively performs a number of polynomial bounded queries as follows.

- **sExtract queries**: When $\mathcal{A}$ issues a query on the signing attribute set A_s, if $A_s \notin W_s^*$, $\mathcal{B}$ randomly selects $\hat{r} \in \mathbb{Z}_p^*$ and calculates the vector $\vec{\gamma} = (\gamma_1, \gamma_2, \cdots, \gamma_{n^*}) \in \mathbb{Z}_p^{n^*}$ where $\gamma_1 = -1$ such that $\vec{\gamma} \cdot \vec{M}_i^* = 0$ for all i where $\rho_s^*(i) \in A_s$. $\mathcal{B}$ implicitly defines $r_s = \hat{r} + \gamma_1 a^q + \gamma_2 a^{q-1} + \cdots + \gamma_{n^*} a^{q-n^*+1}$ and computes $L_s = g^{\hat{r}} \prod_{i=1}^{n^*} (g^{a^{q+1-i}})^{\gamma_i}$, $K_s = g^{\alpha'} g^{a\hat{r}} \prod_{i=2}^{n^*} (g^{a^{q+2-i}})^{\gamma_i}$ and $K_{s,x} = L_s^{f_x} \prod_{j=1}^{n^*} (g^{a^j \cdot \hat{r}} \prod_{\substack{o=1,\cdots,n^* \\ o\neq j}} (g^{a^{q+1+j-o}})^{\gamma_o})^{M_{i,j}^*}$ for any $x \in A_s$. If $\rho_s^*(i) \neq x$ for all i, $\mathcal{B}$ simply sets $K_{s,x} = L_s^{f_x}$.

  Then $\mathcal{B}$ returns the signing key $SK_{A_s} = \{L_s, K_s, \{K_{s,x}\}_{x\in A_s}\}$ to $\mathcal{A}$.

  Correctness:

$$L_s = g^{\hat{r}} \prod_{i=1}^{n^*} (g^{a^{q+1-i}})^{\gamma_i} = g^{r_s},$$

$$\begin{aligned} K_s &= g^{\alpha'} g^{a\hat{r}} \prod_{i=2}^{n^*} (g^{a^{q+2-i}})^{\gamma_i} = g^{\alpha'} g^{a^{q+1}} \cdot g^{a\hat{r}} \cdot g^{-a^{q+1}} \prod_{i=2}^{n^*} (g^{a^{q+2-i}})^{\gamma_i} = g^{\alpha} g^{a\hat{r}} \prod_{i=1}^{n^*} (g^{a^{q+2-i}})^{\gamma_i} \\ &= g^{\alpha} (g^a)^{\hat{r} + \sum_{i=1}^{n^*} (g^{a^{q+1-i}})^{\gamma_i}} = g^{\alpha} g^{ar_s}, \end{aligned}$$

$$\begin{aligned} K_{s,x} &= L_s{}^{f_x} \cdot \prod_{j=1}^{n^*} (g^{a^j \cdot \hat{r}} \prod_{\substack{o=1,\ldots,n^* \\ k\neq j}} (g^{a^{q+1+j-o}})^{\gamma_o})^{M_{i,j}^*} = g^{\hat{r} f_x} \prod_{i=1}^{n^*} (g^{a^{q+1-i}})^{\gamma_i f_x} \cdot \prod_{j=1}^{n^*} (g^{a^j})^{\hat{r} \cdot M_{i,j}^*} \\ &= (g^{f_x} \prod_{j=1}^{n^*} (g^{a^j})^{M_{i,j}^*})^{\hat{r}} \cdot \prod_{i=1}^{n^*} (g^{a^{q+1-i}})^{\gamma_i f_x} = h_x^{\hat{r}} \cdot \prod_{i=1}^{n^*} (h_x{}^{a^{q+1-i}})^{\gamma_i} = h_x^{r_s}. \end{aligned}$$

- **dExtract queries**: When $\mathcal{A}$ issues a query on the decryption attribute set A_d, $\mathcal{B}$ randomly picks $\hat{r} \in \mathbb{Z}_p^*$, sets $r_d = \hat{r} - a^q$ and computes $L_d = g^{\hat{r}} g_q^{-1}$, $K_d = g^{\alpha'} g_1^{\hat{r}}$ and $K_{d,x} = h_x^{\hat{r}} g_q^{-f_x}$ for any $x \in A_d$. Then $\mathcal{B}$ returns the decryption private key $SK_{A_d} = \{L_d, K_d, \{K_{d,x}\}_{x\in A_d}\}$ to $\mathcal{A}$.

  Correctness:

$$L_d = g^{\hat{r}} g_q^{-1} = g^{\hat{r}} g^{-a^q} = g^{r_d},$$

$$K_d = g^{\alpha'} g_1^{\hat{r}} = g^{\alpha'} g_{q+1} g_1^{\hat{r}} g_{q+1}^{-1} = g^{\alpha' + a^{q+1}} g^{a\hat{r} - a^{q+1}} = g^{\alpha} g^{ar_d},$$

$$K_{d,x} = h_x^{\hat{r}} g_q^{-f_x} = h_x^{\hat{r}} (h_x)^{-a^q} = h_x^{r_d}.$$

- **Signcrypt queries**: When $\mathcal{A}$ issues a query on (m, W_e, W_s, A_d, A_s) and the cuckoo filter,
  - If $A_s \notin W_s^*$, $\mathcal{B}$ gets the private key SK_{A_s} by running the **sExtract queries**. Then $\mathcal{B}$ generates ciphertext CT by executing the **Signcrypt** algorithm and returns to $\mathcal{A}$.
  - If $A_s \in W_s^*$, $\mathcal{B}$ performs the following steps: $\mathcal{B}$ randomly chooses $\varphi_i \in \mathbb{Z}_p^l$ and generates a vector $\vec{\varphi} = (-\varphi_1, -\varphi_2, \cdots, -\varphi_l)$ such that $\vec{\varphi} \cdot M_s = -\vec{1}_n$, that is $\sum_{i=1}^{l} \varphi_i \cdot M_{s,i} = -\vec{1}_n$, and $\varphi_i = 0$ for all $i \in [1, l]$, where $\rho_s(i) \notin A_s$. $\mathcal{B}$ sets $C = mY^{\sigma}$, $S_1 = g^{r_s}$ and computes $\vec{j} = (j_1, j_2, \cdots, j_l) = H_1(S_1, W_e, W_s)$. If $F(\vec{j}) = 0$, $\mathcal{B}$ aborts; Otherwise, $\mathcal{B}$ chooses a random number $\sigma' \in \mathbb{Z}_p^*$, sets $\sigma = \sigma' - \frac{a}{F_1(\vec{j})}$ and computes $C' = g^{\sigma'} {g_1}^{-1/F_1(\vec{j})}$, $C'' = g^{(d\mu+d')\sigma'} {g_1}^{-(\mu d+d')/F_1(\vec{j})}$, where $\mu = H_2(C')$. $\mathcal{B}$ randomly chooses $v_2, \cdots, v_n \in \mathbb{Z}_p^*$ and defines $\vec{v} = (\sigma' - \frac{a}{F_1(\vec{j})}, v_2, \cdots, v_n)$ and $\lambda_i = \vec{v} \cdot M_i = (\sigma' - \frac{a}{F_1(\vec{j})}) M_{i,1} + \sum_{i=2}^{l} v_i M_{i,n}$ for all $i \in [1, n]$. $\mathcal{B}$ sets $C_i = {g_1}^{(\sigma' M_{i,1} + \sum_{i=2}^{l} v_i M_{i,n})} \cdot {g_2}^{-M_{i,1}/F_1(\vec{j})} \cdot {h_{\rho(i)}}^{-\sigma'} \cdot {g_1}^{f_{\rho_e(i)}/F_1(\vec{j})}$ for $i \in [1, l]$, $S_2 = g^{\alpha'} g^{ar_s} (\prod_{i=1}^{l} (h_{\rho_s(i)}^{r_s})^{\varphi_i}) \cdot (g_q{}^{F_1(\vec{j})} g^{F_2(\vec{j})})^{\sigma'} \cdot ({g_1}^{-F_2(\vec{j})/F_1(\vec{j})}) \cdot (C'')^{\beta\xi}$, where $\beta = H_3(W_e, W_s, C, C', C'', \{C_i\}_{i\in[1,l]})$. Finally, $\mathcal{B}$ returns the ciphertext $CT = \{C, C', C'', \{C_i\}_{i\in[1,l]}, S_1, S_2, CF\}$ to $\mathcal{A}$.

Correctness:

$$C' = g^{\sigma'} {g_1}^{-1/F_1(\vec{j})} = g^{\sigma' - a/F_1(\vec{j})} = g^{\sigma},$$

$$\begin{aligned} C_i &= {g_1}^{(\sigma' M_{i,1} + \sum_{i=2}^{l} v_i M_{i,n})} \cdot {g_2}^{-M_{i,1}/F_1(\vec{j})} \cdot {h_{\rho_e(i)}}^{-\sigma'} \cdot {g_1}^{f_{\rho_e(i)}/F_1(\vec{j})} \\ &= g^{a(\sigma' M_{i,1} + \sum_{i=2}^{l} v_i M_{i,n})} \cdot (g^a)^{-aM_{i,1}/F_1(\vec{j})} \cdot {h_{\rho_e(i)}}^{-(\sigma' - a/F_1(\vec{j}))} = g^{a\lambda_i} h_{\rho_e(i)}^{-\sigma}, \end{aligned}$$

$$\begin{aligned} S_2 &= g^{\alpha'} g^{ar_s} (\prod_{i=1}^{l} (h_{\rho_s(i)}^{r_s})^{\varphi_i}) \cdot (g_q{}^{F_1(\vec{j})} g^{F_2(\vec{j})})^{\sigma'} \cdot ({g_1}^{-F_2(\vec{j})/F_1(\vec{j})}) \cdot (C'')^{\beta\xi} \\ &= g^{\alpha'} g^{a^{q+1}} g^{ar_s} \cdot (g_q{}^{F_1(\vec{j})} g^{F_2(\vec{j})})^{\sigma'} \cdot g^{-a^{q+1}} \cdot ({g_1}^{-F_2(\vec{j})/F_1(\vec{j})}) \cdot (C'')^{\beta\xi} \\ &= (g^{\alpha} g^{ar_s}) \cdot (\prod_{i=1}^{l} (h_{\rho_s(i)}^{r_s})^{\varphi_i}) \cdot (g_q{}^{F_1(\vec{j})} g^{F_2(\vec{j})})^{\sigma'} \cdot (g_q{}^{F_1(\vec{j})} g^{F_2(\vec{j})})^{-a/F_1(\vec{j})} \cdot (C'')^{\beta\xi} \\ &= K_s \cdot (K_{s,x})^{\varphi_i} \cdot (y_0 \prod_{i=1}^{l} y_i^{j_i})^{\sigma} \cdot (C'')^{\beta\xi}. \end{aligned}$$

- **Unsigncrypt queries**: When $\mathcal{A}$ issues a query on the ciphertext CT, $\mathcal{B}$ computes the decryption private key SK_{A_d} by executing the **dExtract queries**. Then $\mathcal{B}$ generates the message m by executing the **Unsigncrypt** algorithm and returns to $\mathcal{A}$.

Forgery: $\mathcal{A}$ outputs the valid forgery ciphertext $CT^* = \{C^*, C'^*, C''^*, \{C_i^*\}_{i\in[1,l]}, S_1^*, S_2^*, CF\}$ on (m^*, W_e^*, W_s^*). CT^* satisfies the following two conditions:

1. Since $A_d^* \in W_e^*$, the result of the **Unsigncrypt** algorithm is $m^* \neq \perp$;
2. $\mathcal{A}$ never issues the **Signcrypt queries** on (m^*, W_e^*, W_s^*).

Now, $\mathcal{B}$ could provide the methods to solve the q-CDHE problem as follows.

Firstly, $\mathcal{B}$ computes $\vec{j}^* = (j_1^*, j_2^*, \cdots, j_l^*) = H_1(S_1^*, W_e^*, W_s^*)$. If $b_0 + \sum_{i=1}^{l} j_i b_i \neq \eta\pi$, then $\mathcal{B}$ aborts. Otherwise, $F_1(\vec{j}^*) = 0 \bmod p$, $\mathcal{B}$ computes $C^* = m^* Y^{\sigma}$, $C'^* = g^{\sigma}$, $C''^* = g^{(d\mu+d')\sigma}$, $\{C_i^* = g^{a\lambda_i} h_{\rho_e^*(i)}^{\sigma}\}_{i\in[1,l^*]}$, $S_1^* = g^{r_s}$, $S_2^* = g^{\alpha} g^{ar_s} (\prod_{i=1}^{l^*} (h_{\rho_s^*(i)}^{r_s})^{\varphi_i^*}) (y_0 \prod_{i\in[1.l]} y_i^{j_i^*})^{\sigma} \cdot (g^{(d\mu+d')})^{\sigma\xi\beta^*}$, where $\mu^* =$

$H_2(C'^*)$, $\beta^* = H_3(W_e^*, W_s^*, C^*, C'^*, C''^*, \{C_i^* = g^{a\lambda_i} h_{\rho_e^*(i)}^{\sigma}\}_{i\in[1,l^*]})$ and the vector $\vec{\varphi}^* = (-\varphi_1, -\varphi_2, \cdots, -\varphi_{l^*})$ satisfies $\sum_{i=1}^{l^*} \varphi^* \cdot M_{s,i}^* = -\vec{1}_{n^*}$.

Then $\mathcal{B}$ can calculate $\frac{S_2^*}{g^{\alpha}(\prod_{i=1}^{l^*}(S_1^*)^{f_{\rho_s^*(i)}})(C'^*)^{F_2(\vec{j}^*)+(d\mu+d')\xi\beta^*}} = g^{a^{q+1}}$.

Correctness:

$\sum_{i=1}^{l^*} \varphi_i^* \cdot M_{s,i}^* = -\vec{1}_{n^*}$ implies $\sum_{i=1}^{l^*} \varphi_i^* \cdot M_{i,j}^* = \begin{cases} -1, j = 1; \\ 0, if 2 \le j \le n^* \end{cases}$, so $\sum_{i=1}^{l^*}\sum_{j=1}^{l_*} a^j M_{i,j}^* \varphi_i^* r_s = -ar_s$.

$$
\begin{aligned}
&\frac{S_2^*}{g^{\alpha}(\prod_{i=1}^{l^*}(S_1^*)^{f_{\rho_s^*(i)}})(C'^*)^{F_2(\vec{j}^*)+(d\mu+d')\xi\beta^*}} \\
&= \frac{g^{\alpha} g^{ars} \cdot (\prod_{i=1}^{l^*} h_{\rho_s^*(i)}^{r_s\varphi_i^*})(y_0 \prod_{i\in[1,l^*]} y_i^{j_i^*})^{\sigma} \cdot (g^{(d\mu^*+d')})^{\sigma\xi\beta^*}}{g^{\alpha} \cdot (\prod_{i=1}^{l^*}(S_1^*)^{\varphi_i^* f_{\rho_s^*(i)}}) \cdot (C'^*)^{F_2(\vec{j}^*)+(d\mu+d')\xi\beta^*}} \\
&= \frac{g^{\alpha'+a^{q+1}} g^{ars} \cdot (\prod_{i=1}^{l^*} (g^{f_{\rho_s^*(i)}} \prod_{j=1}^{n} g^{a^j M_{i,j}^*})^{\varphi_i^* r_s}) \cdot (g_q^{F_1(\vec{j}^*)} g^{F_2(\vec{j}^*)})^{\sigma} \cdot (g^{\sigma})^{(d+u^* d')\xi\beta^*}}{g^{\alpha} \cdot (\prod_{i=1}^{l^*}(S_1^*)^{\varphi_i^* f_{\rho_s^*(i)}}) \cdot (C'^*)^{F_2(\vec{j}^*)+(d\mu+d')\xi\beta^*}} \\
&= \frac{g^{\alpha'} g^{a^{q+1}} g^{ars} \cdot (\prod_{i=1}^{l^*} (g^{r_s})^{\varphi_i^* f_{\rho_s^*(i)}}) \cdot (\prod_{i=1}^{l^*}\prod_{j=1}^{n} g^{a^j M_{i,j}^* \varphi_i^* r_s}) \cdot (g^{\sigma})^{F_2(\vec{j}^*)+(d\mu+d')\xi\beta^*}}{g^{\alpha} \cdot (\prod_{i=1}^{l^*}(S_1^*)^{\varphi_i^* f_{\rho_s^*(i)}}) \cdot (C'^*)^{F_2(\vec{j}^*)+(d\mu+d')\xi\beta^*}} \\
&= \frac{g^{\alpha'} g^{a^{q+1}} g^{ars} \cdot (\prod_{i=1}^{l^*} (g^{r_s})^{\varphi_i^* f_{\rho_s^*(i)}}) \cdot g^{-ars} \cdot (C'^*)^{F_2(\vec{j}^*)+(d\mu+d')\xi\beta^*}}{g^{\alpha} (\prod_{i=1}^{l^*}(S_1^*)^{\varphi_i^* f_{\rho_s^*(i)}}) \cdot (C'^*)^{F_2(\vec{j}^*)+(d\mu+d')\xi\beta^*}} = g^{a^{q+1}}.
\end{aligned}
$$

In the Forgery phase, $\mathcal{B}$ can successfully simulate without aborting if $b_0 + \sum_{j\in[1,l^*]} m_j^* b_j = \eta\pi$. The probability of this simulation is not abort is $\frac{1}{\eta}\frac{1}{l+1} = \frac{1}{k(l+1)}$. Therefore, the success probability of $\mathcal{B}$ for solving the q-CDHE problem is at least $\varepsilon' = \varepsilon/k(l+1)$.

7. Performance Analysis

The functionality, computation and communication costs of the proposed CP-ABSC scheme are evaluated in this section. We also compare them with other related schemes [20–23].

7.1. Functionality Comparison

The functionality comparisons between the proposed CP-ABSC scheme and other related schemes [20–23] are presented. Let MC be the message confidentiality, CU be the ciphertext unforgeability, CPA be the chosen plaintext attacks, CCA be the chosen ciphertext attack, CMA be the chosen message attack, ROM be the random model and SM be the standard model. Table 1 summarizes the functionality comparison results.

Table 1. Comparison of computation cost.

Scheme	KP/CP	Access Structure	Public Verifiability	MC	CU	Security Model	Privacy-Preserving
[20]	CP	Monotone tree	No	CPA	CMA	ROM	No
[21]	CP	Monotone tree	Yes	CCA	CMA	SM	No
[22]	KP	Threshold policy	No	CCA	CMA	SM	No
[23]	CP	LSSS	Yes	CCA	CMA	SM	No
our	CP	LSSS	Yes	CCA	CMA	SM	Yes

It is clear from Table 1 that only the scheme [22] adopts the threshold policy as access policy which only supports simple predicates. Although the schemes [20,21] support monotone tree policy which can transform into LSSS access policy, the construction of this type of access structure is quite

complicated. The scheme [23] and our proposed scheme support LSSS access structure that has the simpler construction process. In addition, our scheme and the schemes [21,23] can satisfy public verifiability. All schemes realize CCA security and CMA security in the standard model except [20]. In particular, none of these schemes [20–23] could provide the property of privacy-preserving, only our scheme protects the personal privacy of EHR owners.

7.2. Computation Cost

We analyze the computation cost of the proposed CP-ABSC scheme and compare it with that of other related schemes [20–23]. For computation complexity estimation, we define the following time cost for performing the cryptographic operations required in all schemes. Let T_p be the time for performance a pairing, T_m be the time for performance a scale multiplication in $\mathbb{G}$, T_{mt} be the time for performance a scale multiplication in $\mathbb{G}_T$. Other lightweight operations (the arithmetic operation in $\mathbb{Z}_p$, one-way hash function)are not taken into account.

To offer the security level to 80-bit, we adopt the symmetric bilinear pairing $e: \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$, where $\mathbb{G}$ be the multiplicative cyclic group by p, p is 512-bit prime number. The simulation experiment is based on the C++ Pairing-Based Cryptography (PBC) library MIRACL and runs on Intel Core i5-4590, 3.3 GHz CPU, 8 gigabytes memory with Windows 7 environment.

In this paper, we execute the experiment on a common PC, if the experiment were to run in a practical cloud environment, such as EC2 cloud computing service [49], it would actually run faster. The average execution times of T_p, T_m and T_{mt} are listed in Table 2.

Table 2. Time cost of cryptographic operation.

Cryptographic Operation	Execution Time
Bilinear pairing T_p	9.0791
Scalar multiplication in $\mathbb{G}$ T_m	3.7770
Scalar multiplication in $\mathbb{G}_T$ T_{mt}	0.9243

Let l be the number of attributes in attribute space. We summarize the computation costs of the proposed scheme, Wang et al.'s scheme [20], Emura et al.'s scheme [21], Hu et al.'s scheme [22] and Rao et al.'s scheme [23] in Table 3.

Table 3. Comparison of computation cost.

Scheme	Signcrypt	Unsigncrypt
[20]	7.554l+23.5863 ms	38.165l+37.2407 ms
[21]	15.108l+22.2587 ms	54.4746l+27.2373 ms
[22]	22.662l+23.5683 ms	47.2441l ms
[23]	15.108l+8.4783 ms	20.4101l+52.9495 ms
The proposed scheme	18.885l+27.3633 ms	19.0825l+51.4244 ms

In terms of the **Signcrypt** phase, for the computation costs of l attributes, Wang et al.'s scheme [20] requires to execute $(4l + 3)$ scalar multiplication operations in $\mathbb{G}$, two scalar multiplication operations in $\mathbb{G}_T$ and one bilinear pairing operation. Therefore, the total signcryption time is 7.554l+23.5863 ms. Emura et al.'s scheme [21] needs to execute $(6l + 2)$ scalar multiplication operations in $\mathbb{G}$ and one scalar multiplication operation in $\mathbb{G}_T$. Therefore, the total signcryption time is 15.108l+22.2587 ms. Hu et al.'s scheme [22] needs to execute $(4l + 2)$ scalar multiplication operations in $\mathbb{G}$ and one scalar multiplication operation in $\mathbb{G}_T$. Therefore, the total signcryption time is 22.662l+23.5683 ms. Rao et al.'s scheme [23] needs to execute $(5l + 7)$ scalar multiplication operations in $\mathbb{G}$ and one scalar multiplication operation in $\mathbb{G}_T$. Therefore, the total signcryption time is 15.108l+8.4783 ms. The proposed scheme needs to

execute $(2l + 6)$ scalar multiplication operations in $\mathbb{G}$ and one scalar multiplication operation in $\mathbb{G}_T$. Therefore, the total signcryption time is $18.885l$+27.3633 ms.

In terms of the **Unsigncrypt** phase, for the computation costs of l attributes, Wang et al.'s scheme [20] needs to execute $(2l + 1)$ scalar multiplication operations in $\mathbb{G}_T$ and $(4l + 4)$ bilinear pairing operations. Therefore, the total unsigncryption time is $38.165l$+37.2407 ms. Emura et al.'s scheme [21] needs to execute $(6l + 3)$ bilinear pairing operations. Therefore, the total unsigncryption time is $54.4746l$+27.2373 ms. Hu et al.'s scheme [22] needs to execute $2l$ scalar multiplication operations in $\mathbb{G}_T$ and $5l$ bilinear pairing operations. Therefore, the total unsigncryption time is $47.2441l$ ms. Rao et al.'s scheme [23] needs to execute $(3l + 2)$ scalar multiplication operations in $\mathbb{G}$ and $(l + 5)$ bilinear pairing operations. Therefore, the total unsigncryption time is $20.4101l$+52.9495 ms. The proposed scheme needs to execute four scalar multiplication operations in $\mathbb{G}$, l scalar multiplication operations in $\mathbb{G}_T$ and $(2l + 4)$ bilinear pairing operations. Therefore, the total unsigncryption time is $19.0825l$+51.4244 ms.

Figures 4 and 5 clearly illustrate the computation cost of the signcrypt and unsigncrypt phases with increasing number of attributes l, respectively.

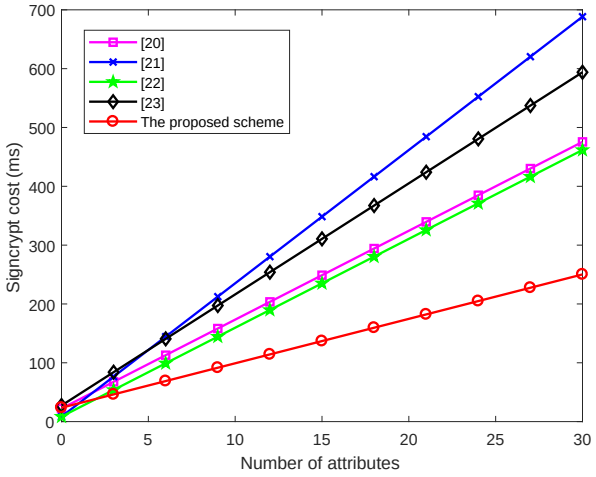

Figure 4. Signcrypt cost with the number of attributes.

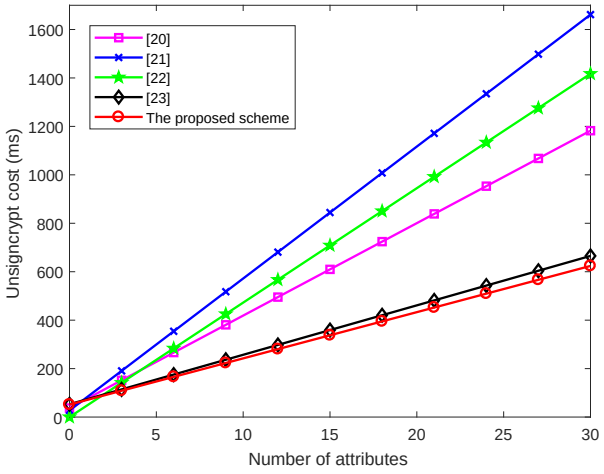

Figure 5. Unsigncrypt cost with the number of attributes.

From Figures 4 and 5, the computation costs in both the signcrypt and unsigncrypt phases rise linearly with the number of attributes in all the schemes. It can be easily see that the proposed scheme's slope is the lowest.

In Figure 4, for $l = 10$, the computation cost of signcrypt is equal to 173.3387, 250.1883, 159.5583, 216.2133 and 99.1263 ms when the schemes [20–23] and the proposed scheme are adopted, respectively. For $l = 30$, the computation cost of signcrypt is equal to 475.4987, 703.4283, 461.7183, 593.9133 and 250.2063 ms when the schemes [20–23] and the proposed scheme are adopted, respectively.

In Figure 5, for $l = 10$, the computation costs of unsigncrypt is equal to 418.8907, 571.9833, 472.441, 257.0505 and 242.2494 ms when the schemes [20–23] and the proposed scheme are adopted, respectively. For $l = 30$, the computation cost of unsigncrypt is equal to 1182.1907, 1661.4753, 1417.323, 665.2525 and 623.8994 ms when the schemes [20–23] and the proposed scheme are adopted, respectively.

According to Figures 4 and 5, we intuitively obtain that the proposed scheme achieves the lowest computation cost with the increase of the number of attributes, especially after adding the cuckoo filter, without increasing extra computation costs in. Therefore, our proposed CP-ABSC scheme is efficient in both the signcrypt and unsigncrypt phase, which has much more advantages than the previous schemes [20–23].

7.3. Communication Cost

We discuss the communication cost of the proposed CP-ABSC scheme with other related schemes [20–23]. Let l be the number of attributes in attribute space, $|\mathbb{G}|$ be the element's length in group $\mathbb{G}$ and $|\mathbb{G}_T|$ be the element's length in group $\mathbb{G}_T$. Since the size of p is 512 bits (64 bytes), therefore the element's size in group $\mathbb{G}$ and $\mathbb{G}_T$ is 512 bits (64 bytes) and 3072 bits (384 bytes), respectively. We also take into account the communication costs of using cuckoo filter. Assume that we use the one-way hash function in cuckoo filter, and its outputs length is 160 bits (20 bytes). When the number of EHR owner's attributes is l, the comparison results on communication cost of these schemes are listed in Table 4.

Table 4. Comparison of communication costs.

Scheme	l **Attributes**
[20]	$256l + 576$ bytes
[21]	$192l + 512$ bytes
[22]	$128l + 576$ bytes
[23]	$128l + 640$ bytes
The proposed scheme	$84l + 640$ bytes

For the communication costs of l attributes, Wang et al.'s scheme [20] includes $(4l + 3)$ the element's length in $\mathbb{G}$ and one the element's length in $\mathbb{G}_T$. Therefore, the total communication cost is $256l$+576 bytes. Emura et al.'s scheme [21] includes $(3l + 2)$ the element's length in $\mathbb{G}$ and one the element's length in $\mathbb{G}_T$. Therefore, the total communication cost is $192l$+512 bytes. Hu et al.'s scheme [22] includes $(2l + 3)$ the element's length in $\mathbb{G}$ and one the element's length in $\mathbb{G}_T$. Therefore, the total communication cost is $128l$+576 bytes. Rao et al.'s scheme [23] includes $(2l + 4)$ the element's length in $\mathbb{G}$ and one the element's length in $\mathbb{G}_T$. Therefore, the total communication cost is $128l$+640 bytes. The proposed scheme includes $(l + 4)$ the element's length in $\mathbb{G}$, one the element's length in $\mathbb{G}_T$ and the outputs length of one-way hash function in cuckoo filter. Therefore, the total communication cost is $84l + 640$ bytes.

Figure 6 demonstrates the relationship between the communication cost and the number of attributes.

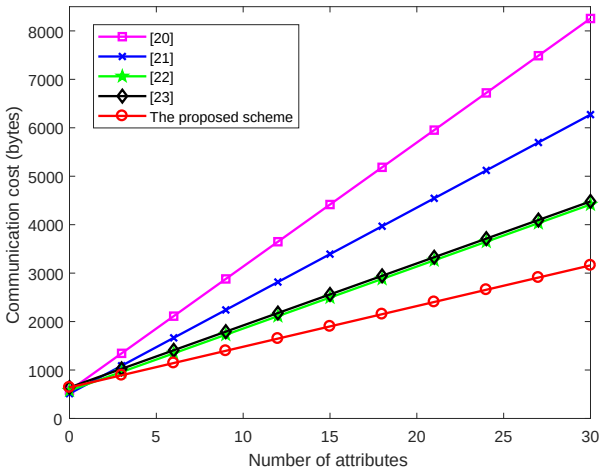

Figure 6. Unsigncrypt cost with the number of attributes.

From Figure 6, the growth of the ciphertext size is linear when the number of attributes increases in all schemes. We could intuitively find out that the communication cost of our proposed scheme is much less than that for other schemes. On the other hand, as Figure 6 shows, when the amount of attributes reaches 30, the communication cost of Wang et al.'s scheme [20], Emura et al.'s scheme [21], Hu et al.'s scheme [22] and Rao et al.'s scheme [23] and the proposed scheme is 7956, 6272, 4416, 4480 and 3100 bytes, respectively. Then the proposed scheme is compared with these schemes [20–23], which can save 61.7%, 57.6%, 28.5%, 29.5% of bandwidth, respectively.

Obviously, although the cuckoo filter is used to hide access policy in this paper, it does not increase communication overhead compared with other schemes. Also, our scheme has the best performance in terms of communication cost in the all five schemes.

In summary, the proposed CP-ABSC scheme achieves low computation and communication cost, which is comparatively more suited to the EHR system.

8. Conclusions

The proposed scheme provides the secure access control of the EHR data as well as prevents the personal privacy information of EHR owners will not be leaked from the LSSS access policy. We show that the proposed scheme is provably security in the standard model under the q-DBDHE assumption and q-CDHE assumption. Detailed performance analysis results indicate that the proposed scheme has lower computation costs and communication overheads than the related schemes. In addition, the proposed scheme protects the EHR owners' sensitive privacy information and is more suitable for EHR system. In the future, we would like to focus on how to design another scheme, such as security and efficient of PPAC scheme without bilinear pairing in EHR system.

Author Contributions: Y.M. and T.Z. conceived of the work, designed the concrete scheme and wrote the paper.

Acknowledgments: This work was supported by the Natural Science Foundation of Shanxi Province (2018JM6081) and the Project of science and technology of Xi'an City (2017088CG/RC051(CADX002)).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Ball, M.; Smith, C.; Bakalar, R.S. Personal health records: Empowering consumers. *J. Healthc. Manag.* **2006**, *21*, 76–86.
2. Hoerbst, A.; Ammenwerth, E. Electronic health records. *Methods Inf. Med.* **2010**, *49*, 320–336. [CrossRef] [PubMed]

3. Badve, O.P.; Gupta, B.B.; Yamaguchi, S. DDoS detection and filtering technique in cloud environment using GARCH model. In Proceedings of the Global Conference on Consumer Electronics (GCCE), Osaka, Japan, 27–30 October 2015; pp. 584–586.
4. Liu, F.; Shu, P.; Jin, H. Gearing resource-poor mobile devices with powerful clouds: architectures, challenges, and applications. *IEEE Wirel. Commun.* **2013**, *20*, 14–22.
5. Alsmirat, M.A.; Jararweh, Y.; Obaidat, I. Internet of surveillance: A cloud supported large-scale wireless surveillance system. *IEEE Wirel. Commun.* **2017**, *73*, 973–992. [CrossRef]
6. Ibraimi, L.; Asim, M.; Petkovi, M. Secure management of personal health records by applying attribute-based encryption. In Proceedings of the International Workshop on Wearable Micro and Nano Technologies for Personalized Health (pHealth), Oslo, Norway, 24–26 June 2009; pp. 71–74.
7. Sun, J.; Fang, Y. Cross-domain data sharing in distributed electronic health record systems. *IEEE Trans. Parallel Distrib. Syst.* **2009**, *6*, 754–764.
8. Akinyele, J.A.; Pagano, M.W.; Green, M.D. Securing electronic medical records using attribute-based encryption on mobile devices. In Proceedings of the ACM Workshop on Security and Privacy in Smartphones and Mobile Devices, Chicago, IL, USA, 17 October 2011; pp. 75–86.
9. Li, M.; Yu, S.; Zheng, Y. Scalable and secure sharing of personal health records in cloud computing using attribute-based encryption. *IEEE Trans. Parallel Distrib. Syst.* **2013**, *24*, 131–143. [CrossRef]
10. Narayan, S.; Gagné, M.; Safavi-Naini, R. Privacy preserving EHR system using attribute-based infrastructure. In Proceedings of the ACM Cloud Computing Security Workshop, Chicago, IL, USA, 8 October 2010; pp. 47–52.
11. Lai, J.; Deng, R.H.; Li, Y. Fully secure cipertext-policy hiding CP-ABE. In Proceedings of the International Conference on Information Security Practice and Experience, Guangzhou, China, 30 May–1 June 2011; pp. 24–39.
12. Liang, X.; Barua, M.; Lu, R. HealthShare: Achieving secure and privacy-preserving health information sharing through health social networks. *Comput. Commun.* **2012**, *35*, 1910–1920. [CrossRef]
13. Lu, R.; Lin, X.; Shen, X. SPOC: A secure and privacy-preserving opportunistic computing framework for mobile-healthcare emergency. *IEEE Trans. Parallel Distrib. Syst.* **2013**, *24*, 614–624. [CrossRef]
14. Liu, Y.; Zhang, Y.; Ling, J. Secure and fine-grained access control on e-healthcare records in mobile cloud computing. *Future Gener. Comp. Sy.* **2018**, *78*, 1020–1026. [CrossRef]
15. Zhou, X.; Liu, J.; Wu, Q. Privacy preservation for outsourced medical data with flexible access control. *IEEE Access.* **2018**, *6*, 14827–14841. [CrossRef]
16. Jiang, S.; Zhu, X.; Wang, L. EPPS: Efficient and privacy-preserving personal health information sharing in mobile healthcare social networks. *Sensors* **2015**, *15*, 22419–22438. [CrossRef] [PubMed]
17. Yang, K.; Han, Q.; Li, H. An efficient and fine-grained big data access control scheme with privacy-preserving policy. *IEEE Internet Things J.* **2017**, *4*, 563–571. [CrossRef]
18. Gagné, M.; Narayan, S.; Safavi-Naini, R. Threshold attribute-based signcryption. In Proceedings of the International Conference on Security and Cryptography for Networks, Amalfi, Italy, 13–15 September 2010; pp. 154–171.
19. Fan, B.; Andersen, D.G.; Kaminsky, M. Cuckoo filter: Practically better than bloom. In Proceedings of the ACM International Conference on Emerging Networking Experiments and Technologies, Sydney, Australia, 2–5 December 2014; pp. 75–88.
20. Wang, C.; Huang, J. Attribute-based signcryption with ciphertext-policy and claim-predicate mechanism. In Proceedings of the International Conference on Computational Intelligence and Security (CIS), Sanya, Hainan, China, 3–4 December 2011; pp. 905–909.
21. Emura, K.; Miyaji, A.; Rahman, M. S. Dynamic attribute-based signcryption without random oracles. *Int. J. Appl. Cryptogr.* **2012**, *2*, 199–211. [CrossRef]
22. Hu, C.; Zhang, N.; Li, H. Body area network security: A fuzzy attribute-based signcryption scheme. *IEEE J. Sel. Areas Commun.* **2013**, *31*, 37–46. [CrossRef]
23. Rao, Y.S. A secure and efficient ciphertext-policy attribute-based signcryption for personal health records sharing in cloud computing. *Future Gener. Comp. Syst.* **2017**, *67*, 133–151. [CrossRef]
24. Akl, S.G.; Taylor, P.D. Cryptographic solution to a problem of access control in a hierarchy. *ACM Trans. Comput. Syst.* **1983**, *1*, 239–248. [CrossRef]

25. Crampton, J.; Farley, N.; Gutin, G. Cryptographic enforcement of information flow policies without public information. In Proceedings of the International Conference on Applied Cryptography and Network Security, New York, NY, USA, 2–5 June 2015; pp. 389–408.
26. Castiglione, A.; De, Santis. A.; Masucci, B. Key indistinguishability versus strong key indistinguishability for hierarchical key assignment schemes. *IEEE Trans. Dependable Secur. Comput.* **2016**, *13*, 451–460. [CrossRef]
27. Castiglione, A.; De, Santis. A.; Masucci, B. Supporting dynamic updates in storage clouds with the Akl-Taylor scheme. *Inf. Sci.* **2017**, *387*, 56–74. [CrossRef]
28. Alderman, J.; Farley, N.; Crampton, J. Tree-Based Cryptographic Access Control. In Proceedings of the European Symposium on Research in Computer Security, Oslo, Norway, 11–15 September 2017; pp. 47–64.
29. Sahai, A.; Waters, B. Fuzzy identity-based encryption. In Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, 22–26 May 2005; pp. 457–473.
30. Goyal, V.; Pandey, O.; Sahai, A. Attribute-based encryption for fine-grained access control of encrypted data. In Proceedings of the ACM conference on Computer and Communications Security, Alexandria, VA, USA, 30 October–3 November 2006; pp. 89–98.
31. Bethencourt, J.; Sahai, A.; Waters, B. Ciphertext-policy attribute-based encryption. In Proceedings of the IEEE Symposium on Security and Privacy, Oakland, CA, USA, 20–23 May 2007; pp. 321–334.
32. Waters, B. Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. In Proceedings of the International Workshop on Public Key Cryptography, Taormina, Italy, 6–9 March 2011; pp. 53–70.
33. Li, J.; Ren, K.; Zhu, B. Privacy-aware attribute-based encryption with user accountability. In Proceedings of the International Conference on Information Security, Pisa, Italy, 7–9 September 2009; pp. 347–362.
34. Zhang, Y.; Chen, X.; Li, J. Anonymous attribute-based encryption supporting efficient decryption test. In Proceedings of the ACM SIGSAC symposium on Information, computer and communications security, Hangzhou, China, 8–10 May 2013; pp. 511–516.
35. Li, J.; Chen, X.; Li, J. Fine-grained access control system based on outsourced attribute-based encryption. In Proceedings of the European Symposium on Research in Computer Security, Egham, UK, 9–13 September 2013; pp. 592–609.
36. Zheng, Y. Digital signcryption or how to achieve cost (signature & encryption) $\ll$ cost (signature)+ cost (encryption). In Proceedings of the Annual International Cryptology Conference, Santa Barbara, CA, USA, 17–21 August 1997; pp. 165–179.
37. Chen, C.; Chen, J.; Lim, H.W. Combined public-key schemes: the case of ABE and ABS. In Proceedings of the International Conference on Provable Security, Chengdu, China, 26–28 September 2012; pp. 53–69.
38. Guo, Z.; Li, M.; Fan, X. Attribute-based ring signcryption scheme. *Secur. Commun. Netw.* **2013**, *6*, 790–796. [CrossRef]
39. Wang, C.J.; Huang, J.S.; Lin, W.L. Security analysis of Gagne et al.'s threshold attribute-based signcryption scheme. In Proceedings of the International Conference on Intelligent Networking and Collaborative Systems (INCoS), Xi'an, China, 9–11 September 2013; pp. 103–108.
40. Han, Y.; Lu, W.; Yang, X. Attribute-based signcryption scheme with non-monotonic access structure. In Proceedings of the International Conference on Intelligent Networking and Collaborative Systems (INCoS), Xi'an, China, 9–11 September 2013; pp. 796–802.
41. Wei, J.; Hu, X.; Liu, W. Traceable attribute-based signcryption. *Secur. Commun. Netw.* **2014**, *7*, 2302–2317. [CrossRef]
42. Pandit, T.; Pandey, S.K.; Barua, R. Attribute-based signcryption: Signer privacy, strong unforgeability and ind-cca2 security in adaptive-predicates attack. In Proceedings of the International Conference on Provable Security, Hong Kong, China, 9–10 October 2014; pp. 274–290.
43. Rao, Y.S.; Dutta, R. Efficient attribute-based signature and signcryption realizing expressive access structures. *Int. J. Inf. Secur.* **2016**, *15*, 81–109. [CrossRef]
44. Liu, J.; Huang, X.; Liu, J.K. Secure sharing of personal health records in cloud computing: Ciphertext-policy attribute-based signcryption. *Future Gener. Comp. Syst.* **2015**, *52*, 67–76. [CrossRef]
45. Wang, D.; Wang, N.; Wang, P. Preserving privacy for free: Efficient and provably secure two-factor authentication scheme with user anonymity. *Inf. Sci.* **2015**, *321*, 162–178. [CrossRef]

46. Wang, D.; Wang, P. Two birds with one stone: Two-factor authentication with security beyond conventional bound. *IEEE Trans. Dependable Secur. Comput.* **2018**, *1*, 708–722. [CrossRef]
47. Bloom, B.H. Space/time trade-offs in hash coding with allowable errors. *Commun. ACM* **1970**, *13*, 422–426. [CrossRef]
48. Pagh, R.; Rodler, F.F. Cuckoo hashing. *J. Algorithms* **2004**, *51*, 122–144. [CrossRef]
49. Wang, D.; Gu, Q.; Cheng, H. The request for better measurement: A comparative evaluation of two-factor authentication schemes. In Proceedings of the ACM on Asia Conference on Computer and Communications Security, Xi'an, China, 30 May–3 June 2016; pp. 475–486.