Privacy-preserving Fusion of IoT and Big Data for E-health

Yang Yang[a,b,c,d], Xianghan Zheng[a,c,d], Wenzhong Guo[a,c,d], Ximeng Liu[a,b], Victor Chang[e]

[a]*College of Mathematics and Computer Science, Fuzhou University, Fuzhou, China.*
[b]*University Key Laboratory of Information Security of Network Systems (Fuzhou University), Fujian Province, China.*
[c]*Fujian Provincial Key Laboratory of Network Computing and Intelligent Information Processing, Fuzhou University, Fuzhou, China*
[d]*Key Laboratory of Spatial Data Mining & Information Sharing, Ministry of Education, Fuzhou, China*
[e]*IBSS, Xi'an Jiaotong-Liverpool University, Suzhou, China.*

Abstract

In this paper, we propose a privacy-preserving e-health system, which is a fusion of Internet-of-things (IoT), big data and cloud storage. The medical IoT network monitors patient's physiological data, which are aggregated to electronic health record (EHR). The medical big data that contains a large amount of EHRs are outsourced to cloud platform. In the proposed system, the patient distributes an IoT group key to the medical nodes in an authenticated way without interaction round. The IoT messages are encrypted using the IoT group key and transmitted to the patient, which can be batch authenticated by the patient. The encrypted EHRs are shared among patient and different data users in a fine-grained access control manner. A novel keyword match based policy update mechanism is designed to enable flexible access policy updating without privacy leakage. Extensive comparison and simulation results demonstrate that the algorithms in the proposed system are efficient. Comprehensive analysis is provided to prove its security.

Email addresses: `yang.yang.research@gmail.com` (Yang Yang), `xianghan.zheng@fzu.edu.cn` (Xianghan Zheng), `guowenzhong@fzu.edu.cn` (Wenzhong Guo), `snbnix@gmail.com` (Ximeng Liu), `victorchang.research@gmail.com` (Victor Chang)

Keywords: Internet-of-things, big data, privacy-preserving, cloud storage, access control, keyword match based policy update

1. Introduction

The rapid development of Internet of Things (IoT) [1, 2] has greatly changed our daily life, especially in the electronic health (e-health) domain [3]. The patients with chronic diseases or severe illnesses are equipped with implanted or on the surface medical sensors to monitor various kinds of physiological data. These medical nodes are accommodated in the e-health IoT network, and the collected physiological data are gathered to patient's gateway device through Internet, which forms patient's electronic health record (EHR). As the data accumulated, the EHRs of the patients become medical big data with high volume, which faces a lot of challenges, such as the information privacy, search, updating and sharing. It is emergent to design a privacy-preserving e-health system for the fusion of IoT and medical big data [4, 5, 6], which handles these challenges.

In the e-health IoT network, the collected physiological data may leak patient's privacy and should to be encrypted to guarantee the confidentiality. The key distribution in IoT is a major issue to be handled such that the patient's gateway device and all the medical nodes share the same IoT symmetric key. The existing schemes requires interaction between the gateway device and the medical nodes to negotiate a secret key, which consumes a large number of transmission and computation costs. Considering the low battery supply and weak computing capability of the tiny medical sensors, it is desirable to generate the IoT group key (among patient and the medical nodes) without key negotiation rounds and at the same ensure the security, such as the source authentication. After the IoT key is distributed, the medical nodes encrypt the health data using the IoT key and transmit them to the patient. To resist impersonation attack, the patient authenticates these IoT ciphertexts to make sure that they are sent by the nodes in his own IoT network. Batch verification method should be designed to accelerate the authenticating speed.

The EHRs in the e-health big data system also requires the privacy protection, and to be shared between patients and the designated data users. Attribute based encryption (ABE) provides an ideal method to realize fine-grained access control and can be adopted in the medical big data system. The system users are assigned with attribute secret key and the patients' EHRs are encrypted with access policy such that only the users with proper attributes secret key are authorized to access to patients' encrypted EHRs.

The policy updating is a big challenge in health big data environment because the encrypted EHRs are outsourced to the medical cloud rather than locally stored. It is not realistic for the patient to download all the EHR ciphertexts with an old access policy, decrypt and re-encrypt them with a new access policy, which brings heavy transmission and computation burden to the patient. On the other hand, the patients hope to make the access policy updating in a finer controlled manner such that only the encrypted EHRs contain certain keywords are updated to the new access policy. This problem is not considered and solved in the existing works.

1.1. Our Contributions

To handle the above challenges, a privacy-preserving e-health system with the fusion of IoT and big data is designed and the main contributions are summarized below.

- ***Anonymity and traceability of patient and medical node***: In the e-health system, the identities of patient and the medical nodes in patient's IoT network are sensitive and may leak the privacy. In this system, anonymous identities are assigned for both patient and medical nodes, which is calculated from their real identities. If an anonymous patient is found dishonest or misbehaving, the trusted authority is capable to trace his real identity. If an anonymous medical node is compromised and utilized to launch attack in patient's IoT network, the patient can also recover the node's real identity.

- ***Authenticated IoT key distribution***: In order to guarantee the confidentiality of the messages transmitted in the health IoT network, the patient generates a symmetric key and sends it to all the medical nodes (in patient's IoT network) in a privacy-preserving way. A key extraction auxiliary message is created by the patient to encapsule the IoT key. Receiving the message, the medical nodes authenticate that whether the auxiliary message is indeed sent by the patient to prevent impersonation attack.

- ***Authenticated IoT ciphertext transmission***: After the IoT key is extracted by the medical nodes, the generated IoT messages are encrypted by the key and sends to the patient. To ensure the reliability of the source, the patient authenticates the IoT ciphertext and then decrypt it. A large amount of IoT ciphertexts may arrive at the same time period, our system also provides a batch verification algorithm to improve the efficiency.

- ***Lightweight fine-grained access control***: The e-health big data are encrypted and stored in a cloud platform. To prevent unauthorized data access, our system designs an expressive and lightweight fine-grained access control mechanism. The patient controls the electronic health record (EHR) encryption procedure, and defines an access policy such that the data users with specific attributes can decrypt patient's medical files. The algorithms in the access control mechanism are lightweight constructions.

- ***Flexible subset keyword match based access policy update***: We design a novel access policy update mechanism based on keyword match. When the patient wants to change the access policy, he does not need to download all his EHR ciphertexts, decrypt them and re-encrypt them with a new access policy. In our system, the patient generates a privacy-preserving policy update query, which is sent to the cloud server. Then, the cloud server updates the match files without knowing the plaintext. A highlight is that the policy update process is keyword match based. If the patient wants to update the access policy of the EHRs that contains certain keywords, the query keywords are embedded in the policy update query. Only the access policy match files that contains these query keywords as subset keywords will be updated.

1.2. Related Work

1.2.1. Security of IoT and Big Data

The Internet-of-things (IoT) network contains a lot of security and privacy concerns. Arias et al. [7] analyze the hardware security of the wearable devices and user privacy in wearable IoT system. The secure storage and forward proxy problem is studied in [8] for the dynamic machine-to-machine IoT application. Mutual authentication protocol is designed in [9] for smart city IoT application, which is constructed based on learning with errors complexity assumption. Zhang et al. [10] suggest a three-factor key agreement protocol for e-health system, which supports dynamic authentication. Another key agreement protocol for multi-gateway IoT network is proposed in [11]. The public auditing and verifiable updating issues are investigated in [12]. The big data security in smart grid system is considered in [13]. Liu et al. [14] design a merkel hash tree to realize public auditing in the big data storage system. A secure deduplication scheme [15] is constructed for encrypted big data.

1.2.2. Searchable Encryption

Searchable encryption is a technology to realize keyword search functor over encrypted data. In 2004, Boneh et al.[16] propose a searchable encryption in the public key setting. Xu et al. [17] study the fuzzy keyword search and construct a concrete probable security scheme. The searchable proxy re-encryption is investigated in [24] to resist post-quantum attack. Wang et al. [19] propose a ranked keyword search scheme over outsourced cloud data, and Cao et al. [20] construct a rank scheme supporting multiple keywords. A dynamic searchable encryption system is suggested by Cash et al. [21] for very-large database. Li et al. [32] studies the quality of protection (QoP) and quality of experience (QoE) issues of searchable encryption in the mobile cloud application. Yang et al. [23] investigate the flexible conjunctive keyword search and time controlled authorization problems and construct a concrete searchable encryption system. Some other research work [24] is proposed to realize flexible keyword search.

1.2.3. Attribute Based Encryption

The concept of attribute-based encryption (ABE) is introduced by Goyal et al. [25] and has received extensive extension [26, 27, 28] since then. Yang et al. propose the sharable and traceable ABE scheme in [29] and study the break-glass access in emergency scenario in [30]. The outsourced policy update problem is investigated in [31], where different types of access policies are taken into consideration. To update linear secret sharing scheme (LSSS) based access policy, the scheme in [31] designs a policy compare algorithm to divide the update policy into three types and disparate methods to update the old policy according the different types. The policy updating method in [31] is utilized to construct other ABE schemes in [32, 33]. The searchable encryption and ABE are integrated in [34, 35] such that the authorized data user (with attributes satisfy the access policy) can utilize a keyword based trapdoor to retrieve on data owner's encrypted files. The dynamic auditing and attribute revocation issues are studied in [36].

1.3. Road Map

The rest of this paper is organized as follows. In Section 2, bilinear paring, hardness assumption and linear secret sharing scheme are introduced. Section 3 describes the system model and security model. In Section 4, we propose the system to fuse the health IoT network and big data storage platform. In Section 5 and 6, the security of the proposed system is analyzed. In Section 7, a comprehensive performance analysis and comparison is given. Section 8 concludes this paper.

2. Preliminary

2.1. Bilinear Pairing and Hardness Assumption

Let G and G_T be cyclic groups and g be a generator of G. The bilinear map $e : G \times G \to G_T$ has the following characteristics: (1) bilinear: $\forall u, v \in G$ and $\forall a, b \in \mathbb{Z}_p$, $e(u^a, v^b) = e(u, v)^{ab}$; (2) non-degenerate: $e(g, g) \neq 1$; (3) computable: e can be efficiently calculated.

Decisional Bilinear Diffie-Hellman (DBDH) Assumption. Let $a, b, c \in Z_p$ be chosen at random, and g be a generator of G. An adversary $\mathcal{A}$ is given a tuple $\vec{y} = (g, g^a, g^b, g^s)$, it is hard for $\mathcal{A}$ to distinguish $e(g, g)^{abs}$ from a random element Z from G_T.

2.2. Linear Secret Sharing Scheme

Definition 1 (Linear Secret Sharing Scheme (LSSS) [37]). A secret-sharing scheme Π over a set of parties $\mathcal{P}$ is called linear (over Z_p) if

- The shares for each party form a vector over Z_p.
- There exists a matrix $\mathbb{A}$ with n_1 rows and n_2 columns called the share-generating matrix for Π. For all $i = 1, \cdots, n_1$, the ith row $\mathbb{A}_i$ of the matrix $\mathbb{A}$ is labeled by a party $\rho(i)$ (ρ is a function from $\{1, \cdots, n_1\}$ to $\mathcal{P}$). Set the column vector $V' = (s, v_2, \cdots, v_n)$, where $s \in Z_p$ is the secret to be shared and $v_2, \cdots, v_n \in Z_p$ are randomly chosen. $\mathbb{A} \cdot V'$ is the vector of n_1 shares of the secret s according to Π and the share $\mathbb{A}_i \cdot V$ belongs to party $\rho(i)$.

Suppose that Π is an LSSS for the access structure Φ. Let $S \in \Phi$ be any authorized set and $I \subset \{1, \cdots, n_1\}$ be defined as $I = \{i : \rho(i) \in S\}$. Then, there exists constants $\{\lambda_i \in Z_p\}_{i \in I}$ such that, if $\{s_i\}_{i \in I}$ are valid shares of any secret s according to Π, then $\sum_{i \in I} \lambda_i s_i = s$ and $\sum_{i \in I} \lambda_i \mathbb{A}_i = (1, 0, \cdots, 0)$. Furthermore, it is shown in [37] that these constants $\{\lambda_i\}_{i \in I}$ can be found in time polynomial in the size of the share-generating matrix $\mathbb{A}$. For unauthorized sets, no such constants exist.

3. System and Security Model

3.1. System Model

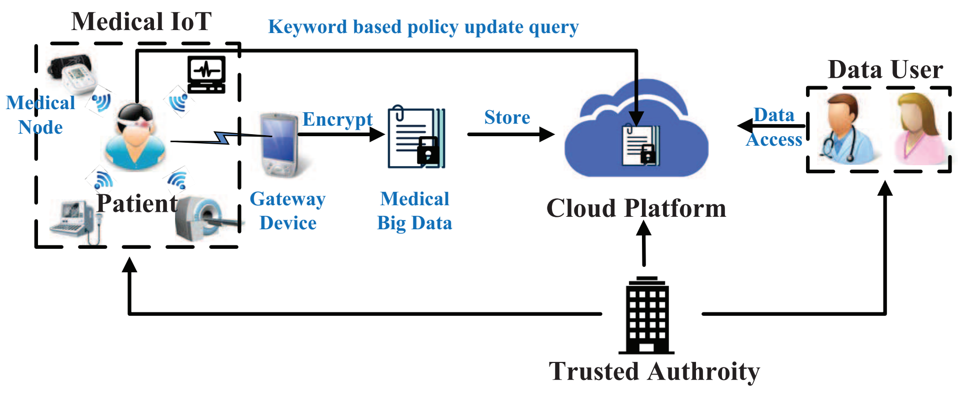

Figure 1: System Model

The system model for the fusion of IoT and big data in this work is depicted in Figure 1. The following roles are involved in the system.

- Trusted authority (TA): is fully trusted in the system, and responsible to generate the public parameter and master secret key. TA is also tasked to create the public/secret key pair for the patients and data users.
- Patient (PA): is monitored by the medical IoT network, and responsible to generate the public/secret key pair for the medical nodes in the network. PA also generates IoT key to ensure the privacy-preserving message transmission in its medical IoT network. The collected data from IoT are aggregated to electronic health record (EHR), which is encrypted by PA with proper access policy such that only the designated set of users are authorized to decrypt the EHRs. PA can also utilizes the keyword match based access policy update mechanism to update the pre-defined access policy of the encrypted EHRs stored in the cloud server.
- Medical node (MN): in the patient's medical IoT network is responsible

to collect the physiological data and encrypts it using the IoT key. Then, MN sends the encrypted IoT message to patient.

- Data User (U): receives the attribute public/secret key from TA and is able to decrypt patient's encrypted EHRs using the attribute secret key if U's attribute set satisfies the EHR's access policy.
- Cloud platform: is responsible to stored the encrypted EHRs. When the keyword match based access policy is received from the patient, the cloud platform runs ciphertext update algorithm to find the match EHRs ciphertexts and update the access policy.

3.2. Security Model

The privacy-preserving system for the fusion of IoT and big data is indistinguishable against chosen plaintext and chosen keyword attach (IND-CPCKA) if there is no probabilistic polynomial time adversary $\mathcal{A}$ can win the following interactive game (between adversary $\mathcal{A}$ and challenger $\mathcal{C}$) with non-negligible advantage.

- *Setup*: The global setup algorithm is executed and the public parameter PP is generated, which is given to $\mathcal{A}$ and the master secret key MSK is kept secret from $\mathcal{A}$.
- *Phase 1.* The following queries are issued by the adversary $\mathcal{A}$.
  - *Patient's secret key query*: $\mathcal{A}$ queries on the patient PA's attribute secret key with attribute set $S = (attr_1 \cdots attr_{n_{PA}})$. $\mathcal{C}$ constructs PA's secret key SK_{PA}, which is sent to $\mathcal{A}$.
  - *Users's secret key query*: $\mathcal{A}$ queries on the user U's attribute secret key with attribute set $S = (attr_1 \cdots attr_{n_U})$. $\mathcal{C}$ constructs U's secret key SK_U, which is sent to $\mathcal{A}$.
  - *Mobile node's secret key query*: $\mathcal{A}$ queries on the mobile node MN's secret key, which belongs to the patient PA's IoT network. $\mathcal{C}$ constructs MN's secret key SK_{MN}, which is sent to $\mathcal{A}$.

  - *Keyword match based policy update query*: $\mathcal{A}$ queries on the patient PA, keyword set $QW = (w_{\sigma_1}, \cdots, w_{\sigma_{l_2}})$ and access policies $(\mathbb{A}, \rho)$, $(\mathbb{A}', \rho')$, $\mathcal{C}$ constructs the policy update query PUQ, which is sent to $\mathcal{A}$.

- *Challenge*: $\mathcal{A}$ sends to $\mathcal{C}$ a challenge access policy $(\mathbb{A}^*, \rho^*)$, a challenge patient identity PA^*, a challenge EHR M^*, two EHR encryption keys $(\Upsilon_0^*, \Upsilon_1^*)$, and two challenge keyword sets (KW_0^*, KW_1^*), where $\mathbb{A}^* \in Z_p^{n_1^* \times n_2^*}$ and ρ^* maps $\mathbb{A}^*$'s rows to attributes. It is required that the secret key of the attribute set S that satisfies the challenge access policy $(\mathbb{A}^*, \rho^*)$ is not queried in phase 1. Moreover, the secret key of PA^* is not queried. $\mathcal{C}$ flips random coins $\mu_1, \mu_2 \in \{0, 1\}$ and the challenge EHR ciphertext CT^* of $(\Upsilon_{\mu_1}^*, KW_{\mu_2}^*)$ is constructed, which is sent to $\mathcal{A}$.

- *Phase 2*: It is the same as in Phase 1 except that the secret key of the attribute set S that satisfies the challenge access policy $(\mathbb{A}^*, \rho^*)$ is not allowed to be queried. In addition, the secret key of PA^* is not allowed to be queried.

- *Guess*: Finally, $\mathcal{A}$ outputs $\mu_1', \mu_2' \in \{0, 1\}$. If $\mu_1' = \mu_1$ and $\mu_2' = \mu_2$, $\mathcal{A}$ wins the game.

3.3. System Requirements

This fusion of IoT and big data system needs to guarantee the following functional and security requirements.

- *Patient anonymity and traceability*: Patient's identity is sensitive in the e-health system and should be hidden from the other system users. The privacy-preserving system needs to provide patient anonymity function. On the other hand, patient anonymous identity should be traceable to prevent misbehave issues.

- *User anonymity and traceability*: The data users include the healthcare staffs, patients' friends and relatives etc. Their identity may leak patients identity and disease related information and needs to be protected by the anonymous mechanism. On the other hand, the anonymous identity should be traceable to prevent misbehave issues.

- *Medical node anonymity and traceability*: The real identity of medical nodes in IoT network may leak the variety of disease of the patient and should be protected by the anonymous mechanism. If a medical node is compromised and utilized to launch attacks, its real identity should be revealed by the trace mechanism.

- *Confidentiality of IoT key*: The IoT key is important and utilized to encrypt the messages sent in the IoT network. The confidentiality of IoT key ensures that the key cannot be recovered by the attackers in the key distribution procedure.

- *Authentication of IoT key distribution*: In the medical IoT network, the key used to encrypt the IoT message will be periodically updated. The key distribution procedure should be authenticated such that the medical nodes can make sure the key update message is indeed sent by its owner (the pre-defined patient).

- *Confidentiality of IoT message*: The medical nodes in IoT network collects vital physiological data of the patient and the IoT messages sent by the medical nodes are sensitive. Thus, the confidentiality of the IoT message should be guaranteed.

- *Authentication of IoT message*: The patient should authenticate that the received encrypted IoT message is indeed sent by the claimed medical node in his own medical IoT network.

- *Secure against replay attack*: The system should guarantee that it can be detected if the previous sent message is replayed by the adversary.

Table 1: Notations

Notation	Description
TA	trusted authority
PA	patient
MN	medical Node of IoT
EHR	electronic health record
PID_{PA}	the pseudo identity of PA
PID_{MN}	the pseudo identity of MN
Σ_{MN}	$\{PID_{MN_1}, \cdots, PID_{MN_n}\}$
m	message sent in IoT
M	EHR in big data system
C_m/CT	ciphertext of m/M
PP/MSK	public parameter/master secret key of the system
PK_{PA}/SK_{PA}	public/secret key of PA
PK_{MN}/SK_{MN}	public/secret key of MN
κ	security parameter
p	a prime number
Z_p^*	$\{1, \cdots, p-1\}$
$i \in [l]$	$i \in \{1, \cdots, l\}$
$b \in_R B$	b is randomly chosen from B
$SEnc/SDec$	secure symmetric encryption/decryption
$\mathcal{K}$	symmetric key space of $SEnc/SDec$
$attr/S$	attribute/attribute set
$(\mathbb{A}, \rho)$	access policy
k	symmetric key for e-health IoT network
KEA	IoT key extraction auxiliary messages
PUQ	policy update query
H_0	secure hash function $H_0 : \{0,1\}^* \to \mathcal{K}$
H_1	secure hash function $H_i : \{0,1\}^* \to Z_p^*$
$KW = (w_1, \cdots, w_{l_1})$	keyword set extracted from the EHR
$QW = (w_1, \cdots, w_{l_2})$	keyword set in the policy update query, $l_2 \leq l_1$

4. The proposed system

In this section, the privacy-preserving fusion of IoT and big data system is proposed and the notations used are defined in Table 1. The fusion of IoT and big data system consists of 13 algorithms, which are indicated in Figures 2, 3, 4 and 5. Next, we explain the mechanisms for the fusion of medical IoT and big data.

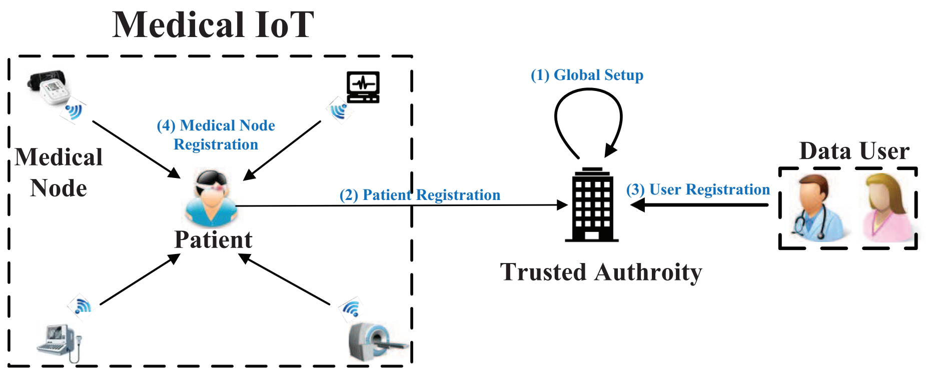

Figure 2: Setup and Registration

As shown in Figure 2, the trusted authority (TA) runs global setup algorithm (Algorithm 1) to generate the public parameter and master secret key for the system. When a patient (PA) registers to the system, TA runs Algorithm 2 to generation the patient's public/secret key pair. When a user (U) registers to the system, TA runs Algorithm 3 to generation the data user's public/secret key pair. When a medical node (MN) in patient PA's healthcare IoT network registers to the system, PA runs Algorithm 4 to generation the medical node's public/secret key pair.

As shown in Figure 3, the patient PA runs e-health IoT key generation algorithm (Algorithm 5) to realize privacy-preserving IoT key distribution. A key extraction auxiliary message KEA is generated and disseminated to the medical nodes in PA's IoT network. Then, these nodes runs authenticated e-health IoT key extraction algorithm (Algorithm 6) to recover the IoT key.

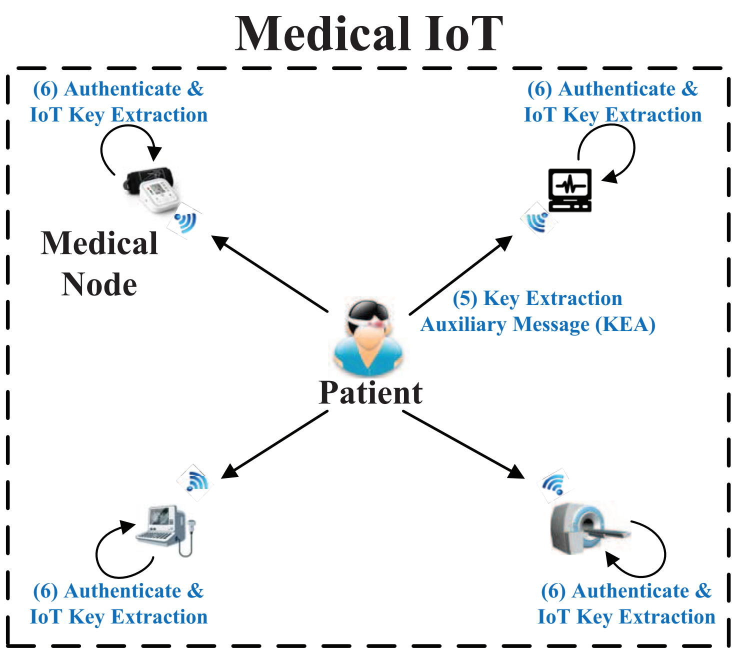
Medical IoT
(6) Authenticate & IoT Key Extraction
(6) Authenticate & IoT Key Extraction
Medical Node
(5) Key Extraction Auxiliary Message (KEA)
Patient
(6) Authenticate & IoT Key Extraction
(6) Authenticate & IoT Key Extraction

Figure 3: E-health IoT Key Generation and Extraction

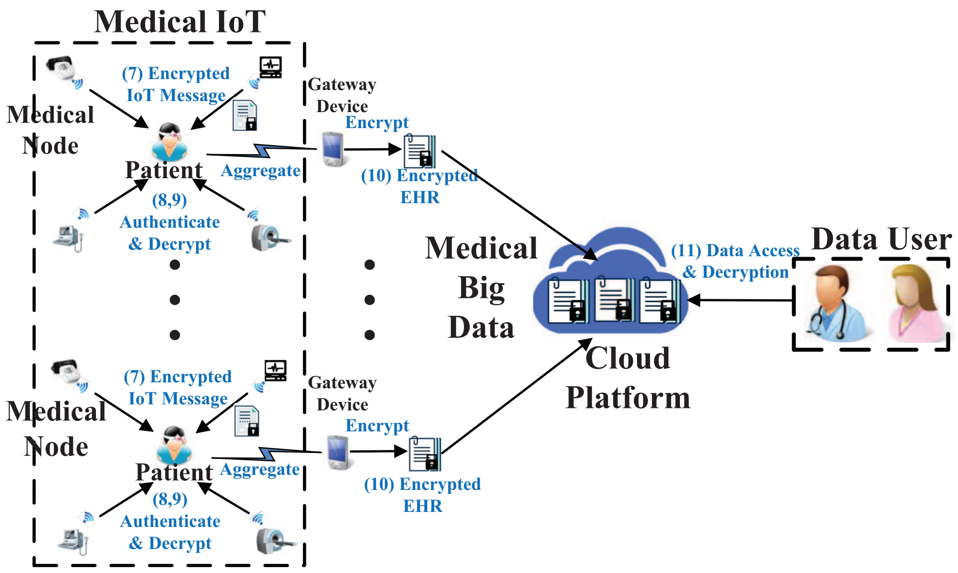

Figure 4: Medical IoT and Big Data Encryption

As shown in Figure 4, the MN runs IoT message encryption algorithm (Algorithm 7), and utilizes the IoT key to encrypt the IoT message. Receiving the encrypted IoT message, the PA runs e-health IoT message verification and decryption algorithm (Algorithm 8) to verify and recover the IoT message. To accelerate the verification, a batch verification algorithm is designed in Algorithm 9. Then, the IoT messages are aggregated to EHR file and encrypted by PA using EHR encryption algorithm 10, where the keywords are extracted and encrypted, and patient defined access policy is exerted to the EHR ciphertext. As the EHRs accumulate, they forms the medical big data system. To save the local storage cost, the medical big data is stored in the cloud platform. The data users with proper attribute secret key is able to access the encrypted EHR data and recover the plaintext using EHR decryption algorithm (Algorithm 11).

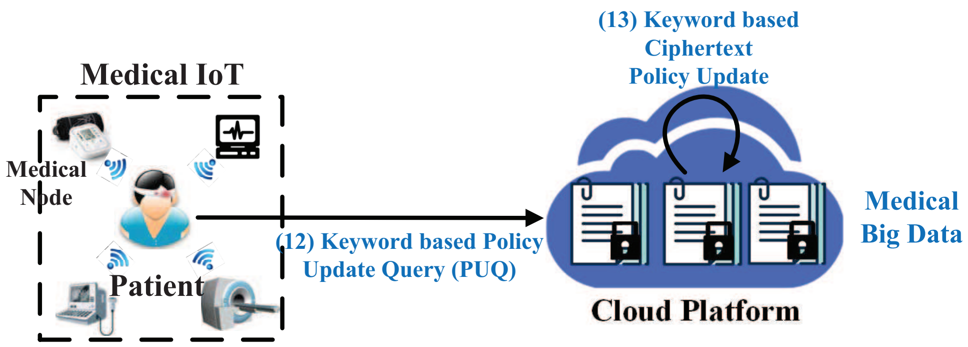
(13) Keyword based Ciphertext Policy Update
Medical IoT
Medical Node
Patient
(12) Keyword based Policy Update Query (PUQ)
Medical Big Data
Cloud Platform

Figure 5: Keyword based Access Policy Update

As shown in Figure 5, if PA wants to update the access policy of the encrypted EHRs, PA runs the keyword match based policy update query algorithm (Algorithm 12) to generate a policy update query PUQ. Receiving the update query, the cloud platform runs the keyword match based ciphertext policy update algorithm (Algorithm 13) to update the ciphertext.

4.1. Global Setup

Given the security parameter κ, TA runs global setup algorithm (shown in Algorithm 1) to generate the public parameter PP and master secret key MSK

for the system. PP is public in the system and MSK is confidentially stored by TA. PP is a default input in the following algorithms, which is omitted to simplify the presentation.

```
Algorithm 1: Global Setup Algorithm
  Input: security parameter $\kappa$.
  Output: $(PP, MSK)$.
1 $TA$ chooses a $\kappa$-bit prime number $p$;
2 Choose bilinear map parameters $(e, G, G_T, g)$, where $g$ is the generator of $G$;
3 Select hash functions $H_0 : \{0,1\}^* \to \mathcal{K}$, $H_1 : \{0,1\}^* \to Z_p^*$;
4 Select symmetric encryption/decryption pair $SEnc/SDec$ with key space $\mathcal{K}$;
5 Choose $\alpha, \beta, a \in_R Z_p^*, g_1 \in G$;
6 Compute $g_2 = g^\beta, \theta = e(g,g), \theta^\alpha = e(g,g)^\alpha$;
7 Set $PP = (g, g_1, g_2, \theta^\alpha)$;
8 Set $MSK = (\alpha, \beta, a)$;
9 Return $(PP, MSK)$.
```

4.2. Patient Registration

A patient PA with attribute set $S = (attr_1, \cdots, attr_{n_{PA}})$ registers to the system. TA runs patient registration algorithm (shown in Algorithm 2) to generate PA's public/secret key pair PK_{PA}/SK_{PA}. TA firstly generates PA's pseudonym identity PID_{PA} using the master secret key MSK and the symmetric encryption algorithm $SEnc$ (Line 1). PA's public key PK_{PA} is constructed in Line 2-4 and secret key SK_{PA} is calculated in Line 5-9.

4.3. User Registration

A user U with attribute set $S = (attr_1, \cdots, attr_{n_U})$ registers to the system, who can be the healthcare staffs, patients' friends and relatives. TA runs user registration algorithm (shown in Algorithm 3) to generate U's public/secret key pair PK_U/SK_U, which is similar to patient registration algorithm (shown in Algorithm 2).

Algorithm 2: PATIENT REGISTRATION ALGORITHM

```
   **Input**: $MSK$, $PA$, $S = (attr_1, \cdots, attr_{n_{PA}})$.
   **Output**: $(PK_{PA}, SK_{PA})$.
1  ta computes $PID_{PA} = SEnc(PA, H_0(\alpha, \beta))$;
2  Select $\alpha_{PA}, \beta_{PA}, \gamma_{PA}, r_{PA} \in_R Z_p^*$;
3  Compute $w_{PA,1} = g^{\alpha_{PA}}, w_{PA,2} = g^{\beta_{PA}}, w_{PA,3} = g^{\gamma_{PA}}$;
4  Set $PK_{PA} = (w_{PA,1}, w_{PA,2}, w_{PA,3})$;
5  Set $d_{PA,1} = \alpha_{PA}, d_{PA,2} = \gamma_{PA}$;
6  Compute $d_{PA,3} = g_1^{\beta + \beta_{PA} \cdot H_1(PID_{PA})}$, $d_{PA,4} = g^{\alpha - a \cdot r_{PA}}$;
7  **for** $i = 1$ *to* $n_{PA}$ **do**
8      Calculate $d_{PA,5,i} = g^{a \cdot r_{PA} \cdot [H_1(attr_i)]^{-1}}$;
9  Set $SK_{PA} = (d_{PA,1}, d_{PA,2}, d_{PA,3}, d_{PA,4}, \{d_{PA,5,i}\}_{i \in [n_{PA}]})$;
10 **Return** $(PK_{PA}, SK_{PA})$.
```

Algorithm 3: USER REGISTRATION ALGORITHM

```
   **Input**: $MSK$, $U$, $S = (attr_1, \cdots, attr_{n_U})$.
   **Output**: $(PK_U, SK_U)$.
1  $TA$ computes $PID_U = SEnc(U, H_0(\alpha, \beta))$;
2  Select $\alpha_U, \beta_U, \gamma_U, r_U \in_R Z_p^*$;
3  Compute $w_{U,1} = g^{\alpha_U}, w_{U,2} = g^{\beta_U}, w_{U,3} = g^{\gamma_U}$;
4  Set $PK_U = (w_{U,1}, w_{U,2}, w_{U,3})$;
5  Set $d_{U,1} = \alpha_U, d_{U,2} = \gamma_U$;
6  Compute $d_{U,3} = g_1^{\beta + \beta_U \cdot H_1(PID_U)}$, $d_{U,4} = g^{\alpha - a \cdot r_U}$;
7  **for** $i = 1$ *to* $n_U$ **do**
8      Calculate $d_{U,5,i} = g^{a \cdot r_U \cdot [H_1(attr_i)]^{-1}}$;
9  Set $SK_U = (d_{U,1}, d_{U,2}, d_{U,3}, d_{U,4}, \{d_{U,5,i}\}_{i \in [n_U]})$;
10 **Return** $(PK_U, SK_U)$.
```

4.4. Medical Node in IoT Registration

The medical node MN in patient PA's healthcare IoT network registers to the system. PA runs medical node in IoT registration algorithm (shown in Algorithm 4) to generate MN's public/secret key pair PK_{MN}/SK_{MN}. PA firstly generates MN's pseudonym identity PID_{MN} using PA's secret key SK_{PA} and the symmetric encryption algorithm $SEnc$ (Line 1). MN's public key PK_{MN} is constructed in Line 2-3 and secret key SK_{MN} is calculated in Line 4-5.

```
Algorithm 4: MEDICAL NODE IN IOT REGISTRATION ALGORITHM
  Input: $SK_{PA}$, $MN$.
  Output: $(PK_{MN}, SK_{MN})$.
1 PA computes $PID_{MN} = SEnc(MN, H_0(SK_{PA}))$;
2 Select $\alpha_{MN} \in_R Z_p^*$;
3 Compute $PK_{MN} = g^{\alpha_{MN}}$;
4 Compute $f_1 = g_1^{\alpha_{PA}+\alpha_{MN}\cdot H_1(PID_{PA},PID_{MN})}$, $f_2 = g_1^{\frac{1}{\alpha_{PA}-H_1(PID_{MN})}}$;
5 Set $SK_{MN} = (f_1, f_2)$;
6 Return $(PK_{MN}, SK_{MN})$.
```

4.5. E-health IoT Key Generation

Suppose $\Sigma_{MN} = \{PID_{MN_1}, \cdots, PID_{MN_n}\}$ is the medical node set of patient PA's health IoT network. To guarantee the privacy-preserving IoT message delivery, PA runs e-health IoT key generation algorithm (shown in Algorithm 5) to generate an e-health IoT key k (Line 2) and the key extraction auxiliary messages $KEA = (b_1, b_2, b_3, b_4, TS_{PA})$ using PA's secret key SK_{PA}, where TS_{PA} is the time stamp to prevent the replay attack. Then, KEA is sent to the medical nodes in Σ_{MN}. It is required that KEA can be authenticated by the MN_i $(1 \le i \le n)$ in Σ_{MN} and the adversary cannot recover k from KEA.

4.6. Authenticated E-health IoT Key Extraction

When the medical node $MN_i \in \Sigma_{MN}$ $(1 \le i \le n)$ receives the IoT key extraction auxiliary messages $KEA = (b_1, b_2, b_3, b_4, TS)$, MN_i runs authenticated

Algorithm 5: E-HEALTH IOT KEY GENERATION ALGORITHM

```
Input: $SK_{PA}$, $\Sigma_{MN} = \{PID_{MN_1}, \cdots, PID_{MN_n}\}$, $TS_{PA}$.
Output: $KEA$.
1 $PA$ selects $\alpha_k, r_k \in_R Z_p^*$;
2 Compute e-health IoT key $k = H_0[e(g, g_1)^{r_k}]$;
3 Compute $b_1 = (w_{PA,1})^{r_k}, b_2 = g^{r_k \prod_{j=1}^{n} H_1(PID_{MN_j})}$;
4 Compute $b_3 = g^{\alpha_k}, b_4 = d_{PA,3} \cdot g_1^{\alpha_k \cdot H_1(PA, \Sigma_{MN}, b_1, b_2, b_3, k, TS_{PA})}$;
5 Set $KEA = (b_1, b_2, b_3, b_4, TS_{PA})$;
6 Return $KEA$.
```

e-health IoT key extraction algorithm (shown in Algorithm 6) to extract the IoT key k. In Line 1, it utilizes its secret key $SK_{MN_i} = (f_1, f_2)$ to extract the IoT key k by calculating

$$k = H_0[e(b_1 \cdot b_2^{1/\prod_{j=1,j\neq i}^{n} H_1(PID_{MN_j})}, f_2)].$$

In Line 2-5, MN_i checks whether KEA is sent by PA, and k is a fresh IoT key generated at time TS_{PA}. MN_i verifies whether the following equation holds

$$e(g, b_4) = e(g_1, g_2 \cdot (w_{PA,2})^{H_1(PID_{PA})} \cdot b_3^{H_1(PA, \Sigma_{MN}, b_1, b_2, b_3, k, TS_{PA})}).$$

If it holds, it indicates that KEA and k pass the verification. Otherwise, KEA is rejected and outputs $\perp$.

Algorithm 6: AUTHENTICATED IOT KEY EXTRACTION ALGORITHM

```
Input: $SK_{MN}$, $KEA$.
Output: $k/\perp$.
1 $MN$ computes e-health IoT key $k = H_0[e(b_1 \cdot b_2^{1/\prod_{j=1,j\neq i}^{n} H_1(PID_{MN_j})}, f_2)]$;
2 if $e(g, b_4) = e(g_1, g_2 \cdot (w_{PA,2})^{H_1(PID_{PA})} \cdot b_3^{H_1(PA, \Sigma_{MN}, b_1, b_2, b_3, k, TS_{PA})})$ then
3     output $k$;
4 else
5     output $\perp$;
6 Return $k/\perp$.
```

4.7. E-health IoT Message Encryption

Algorithm 7: IoT Message Encryption Algorithm

```
Input: $m$, $k$, $SK_{MN}$.
Output: $C_m$.
1 $MN$ selects $r_m \in_R Z_p^*$;
2 Compute $\Phi_{m,0} = SEnc(m, k)$;
3 Compute $\Phi_{m,1} = g^{r_m}, \Phi_{m,2} = f_1 \cdot g_1^{r_m \cdot H_1(PID_{PA}, PID_{MN}, \Phi_{m,0}, TS_{MN})}$;
4 Set $C_m = (\Phi_{m,0}, \Phi_{m,1}, \Phi_{m,2}, TS_{MN})$;
5 Return $C_m$.
```

When the IoT medical node MN collects the physiological message m of patient PA, MN runs e-health IoT message encryption algorithm (shown in 7) to encrypt m and generate the ciphertext $C_m = (\Phi_{m,0}, \Phi_{m,1}, \Phi_{m,2}, TS_{MN})$, where TS_{MN} is the time stamp to generate C_m and C_m is transmitted to PA. It is required that C_m can be authenticated by PA and the adversary cannot recover m from C_m.

4.8. E-health IoT Message Verification and Decryption

The messages in the health IoT network are probably transmitted using the wireless channel, which are prone to be captured, tampered or forged by the adversaries. In order to prevent these attackes, it is important for the patient to verify these encrypted IoT messages. When PA receives the IoT ciphertext $C_m = (\Phi_{m,0}, \Phi_{m,1}, \Phi_{m,2}, TS_{MN})$ from MN, PA runs e-health IoT message verification and decryption algorithm (shown in Algorithm 8) to authenticate C_m and decrypt the underlying message m. In Line 1, PA checks whether C_m is a fresh IoT ciphertext sent by MN, which is generated at time TS_{MN}. PA verifies whether the following equation holds

$$\begin{aligned} & e(g, \Phi_{m,2}) \\ = \quad & e[g_1, w_{PA,1} \cdot (PK_{MN})^{H_1(PID_{PA}, PID_{MN})} \cdot (\Phi_{m,1})^{H_1(PID_{PA}, PID_{MN}, \Phi_{m,0}, TS_{MN})}]. \end{aligned}$$

If it holds (Line 2), it indicates that C_m passes the verification and the algorithm outputs $m = SDec(\Phi_{m,0}, k)$. Otherwise (Line 3-4), C_m is rejected and outputs $\perp$.

Algorithm 8: IoT Message Verification & Decryption Algorithm

```
  Input: $C_m$, $k$, $PK_{PA}$, $PK_{MN}$.
  Output: $m/\perp$.
1 if $e(g, \Phi_{m,2}) = e[g_1, w_{PA,1} \cdot (PK_{MN})^{H_1(PID_{PA},PID_{MN})} \cdot (\Phi_{m,1})^{H_1(PID_{PA},PID_{MN},\Phi_{m,0},TS_{MN})}]$ then
2     $PA$ calculates $m = SDec(\Phi_{m,0}, k)$ and outputs $m$;
3 else
4     output $\perp$;
5 Return $m/\perp$.
```

4.9. E-health IoT Message Batch Verification and Decryption

When PA receives the IoT ciphertext $C_{m_i} = (\Phi_{m_i,0}, \Phi_{m_i,1}, \Phi_{m_i,2}, TS_{MN_i})$ from MN_i for $1 \leq i \leq \tau$, PA runs e-health IoT message batch verification and decryption algorithm (shown in Algorithm 9) to authenticate $(C_{m_1}, \cdots, C_{m_\tau})$ and decrypt the underlying messages $(m_1, \cdots, m_\tau)$.

In Line 1, PA selects random numbers $\delta_1, \cdots, \delta_\tau \in_R Z_p^*$ such that $\Sigma_{i=1}^{\tau}\delta_i = 1 \mod p$. In Line 2, PA checks whether $(C_{m_1}, \cdots, C_{m_\tau})$ are fresh IoT ciphertexts sent by $(MN_1, \cdots, MN_\tau)$, which are generated at time $(TS_{MN_1}, \cdots, TS_{MN_\tau})$, respectively. PA verifies whether the following equation holds

$$\begin{aligned} & e(g, \prod_{i=1}^{\tau}(\Phi_{m_i,2})^{\delta_i}) \\ = \; & e[g_1, w_{PA,1} \prod_{i=1}^{\tau}((PK_{MN_i})^{\delta_i H_1(PID_{PA},PID_{MN_i})} \cdot \\ & (\Phi_{m_i,1})^{\delta_i H_1(PID_{PA},PID_{MN_i},\Phi_{m_i,0},TS_{MN_i})})] \end{aligned}$$

If it holds (Line 3-5), it indicates that $(C_{m_1}, \cdots, C_{m_\tau})$ pass the verification and the algorithm outputs $m_i = SDec(\Phi_{m_i,0}, k)$ for $1 \leq i \leq \tau$. Otherwise (Line 6-7), the algorithm outputs $\perp$.

Algorithm 9: IoT Message Batch Verification & Decryption Algorithm

```
  Input: $(C_{m_1}, \cdots, C_{m_\tau})$, $k$, $PK_{PA}$, $PK_{MN}$.
  Output: $(m_1, \cdots, m_\tau)/\perp$.
1 Select $\delta_1, \cdots, \delta_\tau \in_R Z_p^*$ such that $\Sigma_{i=1}^{\tau}\delta_i = 1 \mod p$;
2 if
      $$e(g, \prod_{i=1}^{\tau}(\Phi_{m_i,2})^{\delta_i}) = e[g_1, w_{PA,1}\prod_{i=1}^{\tau}((PK_{MN_i})^{\delta_i H_1(PID_{PA}, PID_{MN_i})} \cdot (\Phi_{m_i,1})^{\delta_i H_1(PID_{PA}, PID_{MN_i}, \Phi_{m_i,0}, TS_{MN_i})})]$$
   then
3     for $i = 1$ to $\tau$ do
4         $PA$ calculates $m_i = SDec(\Phi_{m_i,0}, k)$;
5     Output $(m_1, \cdots, m_\tau)$;
6 else
7     Output $\perp$;
8 Return $(m_1, \cdots, m_\tau)/\perp$.
```

4.10. Patient Controlled EHR Encryption

After the IoT messages $(m_1, \cdots, m_\tau)$ are received by the patient PA, they are aggregated to an EHR file M. PA extracts a keyword set $KW = (w_1, \cdots, w_{l_1})$ to describe M and an access policy $(\mathbb{A}, \rho)$ to designate the permitted data visitors, where $\mathbb{A} \in Z_p^{n_1 \times n_2}$ is a matrix and the function ρ maps $\mathbb{A}$'s rows to attributes.

The EHR encryption algorithm (shown in Algorithm 10) is executed by the patient PA to generate the ciphertext CT of the EHR file M and the keyword set $KW = (w_1, \cdots, w_{l_1})$. In Line 1-2, the randomly element $\Upsilon \in_R G_T$ is selected, $H_0(\Upsilon)$ is the symmetric encryption key of the EHR file M and the EHR ciphertext is C_M. Υ is encrypted in $C_0 = \Upsilon \cdot \theta^{\alpha s}$. In Line 3-6, the attribute policy $(\mathbb{A}, \rho)$ is encrypted in $\{C_{2,i}\}_{i \in [n_1]}$ using the LSSS mechanism. In Line 7-9, the keywords in KW are encrypted in $\{C_{3,j}\}_{j \in \{0, \cdots, l_1\}}$ such that flexible keyword match query is enabled in the next subsection. In Line 10, the ciphertext is set to be $CT = ((\mathbb{A}, \rho), C_M, C_0, C_1, \{C_{2,i}\}_{i \in [n_1]}, \{C_{3,j}\}_{j \in \{0, \cdots, l_1\}}, C_4)$.

Algorithm 10: EHR ENCRYPTION ALGORITHM

```
   Input: $M$, $SK_{PA}$, $KW = (w_1, \cdots, w_{l_1})$, $(\mathbb{A}, \rho)$, where $\mathbb{A} \in Z_p^{n_1 \times n_2}$.
   Output: $CT$.
1  $PA$ selects $s, r \in_R Z_p^*$ and $\Upsilon \in_R G_T$;
2  Calculate $C_M = SEnc(M, H_0(\Upsilon)), C_0 = \Upsilon \cdot \theta^{\alpha s}, C_1 = g^s, C_4 = \theta^r$;
3  Select $v_2, \cdots, v_{n_2} \in_R Z_p^*$ and set $V = (s, v_2, \cdots, v_{n_2})^\top$;
4  for $i = 1$ to $n_1$ do
5      Calculate $s_i = \mathbb{A}_i \cdot V$;
6      Calculate $C_{2,i} = \rho(i) \cdot s_i \cdot \gamma_{PA}^{-1}$;
7  Select a polynomial $\Gamma(x) = a_{l_1} x^{l_1} + a_{l_1 - 1} x^{l_1 - 1} + \cdots + a_1 x + a_0$ such that
   $\alpha_{PA} H(w_1), \cdots, \alpha_{PA} H(w_{l_1})$ are the $l_1$ roots of the equation $\Gamma(x) = 1$;
8  for $j = 0$ to $l_1$ do
9      Calculate $C_{3,j} = r \cdot a_j \cdot \beta_{PA}^{-1}$;
10 Set $CT = ((\mathbb{A}, \rho), C_M, C_0, C_1, \{C_{2,i}\}_{i \in [n_1]}, \{C_{3,j}\}_{j \in \{0, \cdots, l_1\}}, C_4)$;
11 Return $CT$.
```

4.11. EHR Decryption

When the data user U with attribute set S queries to decrypt patient PA's EHR ciphertext CT with access policy $(\mathbb{A}, \rho)$, U runs the EHR decryption algorithm (shown in Algorithm 11) to recover the plaintext M. If S satisfies $(\mathbb{A}, \rho)$, U utilizes his secret key SK_U and LSSS scheme to decrypt Υ and recover M. Otherwise, the algorithm outputs $\perp$.

```
Algorithm 11: EHR Decryption Algorithm
Input: $PK_{PA}$, $SK_U$ with attribute set $S$, $CT$ with access policy $(\mathbb{A}, \rho)$.
Output: $M/\perp$.
1 if $S$ satisfies $(\mathbb{A}, \rho)$ then
2     Data user utilizes the LSSS scheme to find $\{\lambda_i \in Z_p\}_{i\in[n_1]}$ such that
      $\sum_{i\in[n_1]} \lambda_i \mathbb{A}_i = (1, 0, \cdots, 0)$;
3     Calculate $\Upsilon = C_0/[e(C_1, d_{U,4}) \cdot e(w_{PA,3}, \prod_{i\in[n_1]}(d_{U,5,i})^{C_{2,i}\cdot\lambda_i})]$;
4     Calculate $M = SDec(C_M, H_0(\Upsilon))$;
5 else
6     Output $\perp$;
7 Return $M/\perp$.
```

4.12. Keyword Match based Policy Update Query

If the patient PA wants to update the access policy of the EHR ciphertext that are stored in the e-health big data system, he runs the keyword match based policy update query algorithm (shown in Algorithm 12) to generate a policy update query PUQ, which is submitted to the cloud platform.

Suppose that the original access policy is $(\mathbb{A}, \rho)$ with $\mathbb{A} \in Z_p^{n_1\times n_2}$ and the update access policy is $(\mathbb{A}', \rho')$ with $\mathbb{A}' \in Z_p^{n'_1\times n'_2}$. In the keyword match based policy update mechanism, the patient PA designates a query keyword set $QW = (w_{\sigma_1}, \cdots, w_{\sigma_{l_2}})$. Only the ciphertext with keyword set $KW = (w_1, \cdots, w_{l_1})$ that satisfies $QW \subseteq KW$ will be updated.

In Line 1-4, the update attribute policy $(\mathbb{A}', \rho')$ is encrypted in $\{C'_{2,i}\}_{i\in[n'_1]}$ using the LSSS mechanism. In Line 5-6, the query keywords in QW are encrypted in $\{q_j\}_{j\in\{0,\cdots,l_2\}}$. In Line 7, the policy update query is set to be $PUQ = ((\mathbb{A}, \rho), (\mathbb{A}', \rho'), \{C'_{2,i}\}_{i\in[n'_1]}, \{q_j\}_{j\in\{0,\cdots,l_2\}})$.

4.13. Keyword Match based Ciphertext Policy Update

Receiving the policy update query PUQ from patient PA, the cloud platform runs the keyword match based ciphertext policy update algorithm (shown in Algorithm 13) to update the ciphertext.

Algorithm 12: KEYWORD MATCH BASED POLICY UPDATE QUERY ALGORITHM

```
Input: $SK_{PA}$, $QW = (w_{\sigma_1}, \cdots, w_{\sigma_{l_2}})$, $(\mathbb{A}, \rho)$, $(\mathbb{A}', \rho')$, where $\mathbb{A} \in Z_p^{n_1 \times n_2}$,
        $\mathbb{A}' \in Z_p^{n_1' \times n_2'}$, $l_2 \leq l_1$.
Output: policy update query $PUQ$.
1 $PA$ selects $v_2', \cdots, v_{n_2'}' \in_R Z_p^*$ and sets $V' = (s, v_2', \cdots, v_{n_2'}')^\top$;
2 for $i = 1$ to $n_1'$ do
3     Calculate $s_i' = \mathbb{A}_i' \cdot V'$;
4     Calculate $C_{2,i}' = \rho'(i) \cdot s_i' \cdot \gamma_{PA}^{-1}$;
5 for $j = 0$ to $l_2$ do
6     Calculate $q_j = (l_2 \cdot \gamma_{PA})^{-1} (\alpha_{PA})^j \sum_{k=1}^{l_2} H(w_{\sigma_k})^j$;
7 Set $PUQ = ((\mathbb{A}, \rho), (\mathbb{A}', \rho'), \{C_{2,i}'\}_{i \in [n_1']}, \{q_j\}_{j \in \{0, \cdots, l_2\}})$;
8 Return $PUQ$.
```

Suppose the ciphertext CT has access policy $(\mathbb{A}, \rho)$ and keyword set $KW = (w_1, \cdots, w_{l_1})$, and PUQ has update access policy $(\mathbb{A}', \rho')$ and query keyword set $QW = (w_{\sigma_1}, \cdots, w_{\sigma_{l_2}})$. In Line 1-2, the cloud platform tests whether $QW \subseteq KW$ by verifying whether the equation $e(w_{PA,3}, w_{PA,2})^{\sum_{j=0}^{l_1} (C_{3,j} \cdot q_j)} = C_4$ holds. If it holds, the updated ciphertext is

$$CT' = ((\mathbb{A}, \rho), (\mathbb{A}', \rho'), C_M, C_0, C_1, \{C_{2,i}'\}_{i \in [n_1']}, \{C_{3,j}\}_{j \in \{0, \cdots, l_1\}}, C_4).$$

Otherwise, CT does not satisfy the update requirement. Cloud platform sets $CT' = CT$ to indicate that the ciphertext is not updated (Line 3-4).

5. Correctness Analysis

The correctness of the algorithms in this system are analyzed below.

5.1. Correctness of Authenticated IoT Key Extraction Algorithm

Since $b_1 = (w_{PA,1})^{r_k}$, $b_2 = g^{r_k \prod_{j=1}^{n} H_1(PID_{MN_j})}$, $f_2 = g_1^{\frac{1}{\alpha_{PA} - H_1(PID_{MN})}}$, $w_{PA,1} = g^{\alpha_{PA}}$ and $k = H_0[e(g, g_1)^{r_k}]$, we can deduce that

$$H_0[e(b_1 \cdot b_2^{1/\prod_{j=1, j \neq i}^{n} H_1(PID_{MN_j})}, f_2)]$$

Algorithm 13: KEYWORD MATCH BASED CIPHERTEXT POLICY UPDATE ALGORITHM

```
  Input: $CT$ with access policy $(\mathbb{A}, \rho)$ & keyword set $KW = (w_1, \cdots, w_{l_1})$,
  $PUQ$ with access policy $(\mathbb{A}', \rho')$ & query keyword set $QW = (w_{\sigma_1}, \cdots, w_{\sigma_{l_2}})$.
  Output: $CT'$.
1 if $e(w_{PA,3}, w_{PA,2})^{\sum_{j=0}^{l_1}(C_{3,j}\cdot q_j)} = C_4$ then
2     Cloud sets
      $CT' = ((\mathbb{A}, \rho), (\mathbb{A}', \rho'), C_M, C_0, C_1, \{C'_{2,i}\}_{i\in[n'_1]}, \{C_{3,j}\}_{j\in\{0,\cdots,l_1\}}, C_4)$;
3 else
4     Cloud sets $CT' = CT$;
5 Return $CT'$.
```

$$\begin{aligned}
&= H_0[e((w_{PA,1})^{r_k}\cdot(g^{r_k\prod_{j=1}^{n}H_1(PID_{MN_j})})^{1/\prod_{j=1,j\neq i}^{n}H_1(PID_{MN_j})}, g_1^{\frac{1}{\alpha_{PA}-H_1(PID_{MN})}})]\\
&= H_0[e(g^{r_k\cdot\alpha_{PA}}\cdot g^{r_k\cdot H_1(PID_{MN_i})}, g_1^{\frac{1}{\alpha_{PA}-H_1(PID_{MN})}})]\\
&= H_0[e(g, g_1)^{r_k}] = k.
\end{aligned}$$

Since $b_3 = g^{\alpha_k}$, $b_4 = d_{PA,3}\cdot g_1^{\alpha_k\cdot H_1(PA,\Sigma_{MN},b_1,b_2,b_3,k,TS_{PA})}$, $w_{PA,2} = g^{\beta_{PA}}$ and $d_{PA,3} = g_1^{\beta+\beta_{PA}\cdot H_1(PID_{PA})}$, we can deduce that

$$\begin{aligned}
&e(g, b_4)\\
=\;&e(g, d_{PA,3}\cdot g_1^{\alpha_k\cdot H_1(PA,\Sigma_{MN},b_1,b_2,b_3,k,TS_{PA})})\\
=\;&e(g, g_1^{\beta+\beta_{PA}\cdot H_1(PID_{PA})}\cdot g_1^{\alpha_k\cdot H_1(PA,\Sigma_{MN},b_1,b_2,b_3,k,TS_{PA})})\\
=\;&e(g_1, g^{\beta+\beta_{PA}\cdot H_1(PID_{PA})}\cdot g^{\alpha_k\cdot H_1(PA,\Sigma_{MN},b_1,b_2,b_3,k,TS_{PA})})\\
=\;&e(g_1, g_2\cdot(w_{PA,2})^{H_1(PID_{PA})}\cdot b_3^{H_1(PA,\Sigma_{MN},b_1,b_2,b_3,k,TS_{PA})}).
\end{aligned}$$

5.2. Correctness of IoT Message Verification & Decryption Algorithm

Since $\Phi_{m,1} = g^{r_m}$, $\Phi_{m,2} = f_1\cdot g_1^{r_m\cdot H_1(PID_{PA},PID_{MN},\Phi_{m,0},TS_{MN})}$, $w_{PA,1} = g^{\alpha_{PA}}$, $f_1 = g_1^{\alpha_{PA}+\alpha_{MN}\cdot H_1(PID_{PA},PID_{MN})}$ and $PK_{MN} = g^{\alpha_{MN}}$, we can deduce that

$$e(g, \Phi_{m,2})$$

$$
\begin{aligned}
&= e(g, f_1 \cdot g_1^{r_m \cdot H_1(PID_{PA}, PID_{MN}, \Phi_{m,0}, TS_{MN})}) \\
&= e(g, g_1^{\alpha_{PA} + \alpha_{MN} \cdot H_1(PID_{PA}, PID_{MN})} \cdot g_1^{r_m \cdot H_1(PID_{PA}, PID_{MN}, \Phi_{m,0}, TS_{MN})}) \\
&= e(g_1, g^{\alpha_{PA} + \alpha_{MN} \cdot H_1(PID_{PA}, PID_{MN})} \cdot g^{r_m \cdot H_1(PID_{PA}, PID_{MN}, \Phi_{m,0}, TS_{MN})}) \\
&= e[g_1, w_{PA,1} \cdot (PK_{MN})^{H_1(PID_{PA}, PID_{MN})} \cdot (\Phi_{m,1})^{H_1(PID_{PA}, PID_{MN}, \Phi_{m,0}, TS_{MN})}].
\end{aligned}
$$

5.3. Correctness of IoT Message Batch Verification and Decryption Algorithm

Since $\Phi_{m_i,2} = f_1 \cdot g_1^{r_{m_i} \cdot H_1(PID_{PA}, PID_{MN_i}, \Phi_{m_i,0}, TS_{MN_i})}$, $w_{PA,1} = g^{\alpha_{PA}}$, $PK_{MN_i} = g^{\alpha_{MN_i}}$, $f_1 = g_1^{\alpha_{PA} + \alpha_{MN_i} \cdot H_1(PID_{PA}, PID_{MN_i})}$ and $\Phi_{m_i,1} = g^{r_{m_i}}$, we can deduce that

$$
\begin{aligned}
& e(g, \prod_{i=1}^{\tau} (\Phi_{m_i,2})^{\delta_i}) \\
= \; & e[g, \prod_{i=1}^{\tau} (f_1 \cdot g_1^{r_{m_i} \cdot H_1(PID_{PA}, PID_{MN_i}, \Phi_{m_i,0}, TS_{MN_i})})^{\delta_i}] \\
= \; & e[g, \prod_{i=1}^{\tau} (g_1^{\alpha_{PA} + \alpha_{MN_i} \cdot H_1(PID_{PA}, PID_{MN_i})} \\
& \cdot g_1^{r_m \cdot H_1(PID_{PA}, PID_{MN_i}, \Phi_{m_i,0}, TS_{MN_i})})^{\delta_i}] \\
= \; & e[g_1, \prod_{i=1}^{\tau} (g^{\alpha_{PA} + \alpha_{MN_i} \cdot H_1(PID_{PA}, PID_{MN_i})} \\
& \cdot g^{r_{m_i} \cdot H_1(PID_{PA}, PID_{MN_i}, \Phi_{m_i,0}, TS_{MN_i})})^{\delta_i}] \\
= \; & e[g_1, \cdot \prod_{i=1}^{\tau} (w_1^{\delta_i} \cdot (PK_{MN_i})^{\delta_i \cdot H_1(PID_{PA}, PID_{MN_i})} \\
& \cdot (\Phi_{m_i,1})^{\delta_i \cdot H_1(PID_{PA}, PID_{MN_i}, \Phi_{m_i,0}, TS_{MN_i})})] \\
= \; & e[g_1, w_{PA,1} \cdot \prod_{i=1}^{\tau} ((PK_{MN_i})^{\delta_i \cdot H_1(PID_{PA}, PID_{MN_i})} \\
& \cdot (\Phi_{m_i,1})^{\delta_i \cdot H_1(PID_{PA}, PID_{MN_i}, \Phi_{m_i,0}, TS_{MN_i})})].
\end{aligned}
$$

5.4. Correctness of Keyword Match Based Ciphertext Policy Update Algorithm

Since $C_{3,j} = r \cdot a_j \cdot \beta_{PA}^{-1}$, $q_j = (l_2 \cdot \gamma_{PA})^{-1} (\alpha_{PA})^j \sum_{k=1}^{l_2} H(w_k)^j$, $w_{PA,2} = g^{\beta_{PA}}$ and $w_{PA,3} = g^{\gamma_{PA}}$, we can deduce that

$$
\begin{aligned}
& e(w_{PA,3}, w_{PA,2})^{\sum_{j=0}^{l_1} (C_{3,j} \cdot q_j)} \\
= \; & e(g^{\gamma_{PA}}, g^{\beta_{PA}})^{\sum_{j=0}^{l_1} (r \cdot a_j \cdot \beta_{PA}^{-1}) \cdot (l_2 \cdot \gamma_{PA})^{-1} (\alpha_{PA})^j \sum_{k=1}^{l_2} H(w_k)^j}
\end{aligned}
$$

$$\begin{aligned}
&= e(g,g)^{r\cdot l_2^{-1}\cdot\sum_{j=0}^{l_1}\cdot a_j\cdot(\alpha_{PA})^j\sum_{k=1}^{l_2}H(w_k)^j} \\
&= e(g,g)^{r\cdot l_2^{-1}\cdot\sum_{k=1}^{l_2}(\sum_{j=0}^{l_1}\cdot a_j\cdot(\alpha_{PA}H(w_k))^j)} \\
&= e(g,g)^{r\cdot l_2^{-1}\cdot\sum_{k=1}^{l_2}\Gamma(\alpha_{PA}H(w_k))} \\
&= e(g,g)^{r\cdot l_2^{-1}\cdot\sum_{k=1}^{l_2}1} = e(g,g)^r = C_4.
\end{aligned}$$

5.5. Correctness of EHR Decryption Algorithm

Since $C_1 = g^s$, $C_{2,i} = \rho(i)\cdot s_i\cdot\gamma_{PA}^{-1}$, $w_{PA,3} = g^{\gamma_{PA}}$, $d_{U,4} = g^{\alpha - a\cdot r_U}$ and $d_{U,5,i} = g^{a\cdot r_U\cdot H_1(attr_i)}$, we can deduce that

$$\begin{aligned}
& e(C_1, d_{U,4})\cdot e(w_{PA,3}, \prod_{i\in[n_1]}(d_{U,5,i})^{C_{2,i}\cdot\lambda_i}) \\
= \;& e(g^s, g^{\alpha - a\cdot r_U})\cdot e(g^{\gamma_{PA}}, \prod_{i\in[n_1]}(g^{a\cdot r_U\cdot[H_1(attr_i)]^{-1}})^{\rho(i)\cdot s_i\cdot\gamma_{PA}^{-1}\cdot\lambda_i}) \\
= \;& e(g^s, g^{\alpha - a\cdot r_U})\cdot e(g^{\gamma_{PA}}, g^{a\cdot r_U\cdot\gamma_{PA}^{-1}\cdot\sum_{i\in[n_1]}([H_1(attr_i)]^{-1}\cdot\rho(i)\cdot s_i\cdot\lambda_i)}) \\
= \;& e(g^s, g^{\alpha - a\cdot r_U})\cdot e(g, g^{a\cdot r_U\cdot s}) = \theta^{\alpha s}.
\end{aligned}$$

6. Security Proof

Theorem 6.1. *This system for the fusion of IoT and big data is indistinguishable against chosen plaintext and chosen keyword attack (IND-CPCKA) assuming that the decisional bilinear Diffie-Hellman (DBDH) assumption is intractable.*

Proof. Suppose that the adversary $\mathcal{A}$ is given a tuple (g, g^a, g^b, g^s, Z) as an instance of the DBDH problem, where Z is either $e(g,g)^{abs}$ or a random number in G_T. The interactive game between $\mathcal{A}$ and $\mathcal{C}$ proceeds as below.

- *Setup.* $\mathcal{C}$ chooses $\alpha', \beta \in_R Z_p^*$, $g_1 \in_R G$ and implicitly sets $\alpha = \alpha' + ab$. Compute $g_2 = g^\beta, \theta^\alpha = e(g^a, g^b)\cdot e(g,g)^{\alpha'} = e(g,g)^\alpha$. $\mathcal{C}$ sends the public parameter $PP = (g, g_1, g_2, \theta^\alpha)$ to $\mathcal{A}$.
- *Phase 1.* The following queries are issued by the adversary $\mathcal{A}$.

– *Patient's secret key query*: $\mathcal{A}$ queries on the patient PA's attribute secret key with attribute set $S = (attr_1 \cdots attr_{n_{PA}})$. $\mathcal{C}$ computes $PID_{PA} = SEnc(PA, H_0(\alpha, \beta))$, selects $\alpha_{PA}, \beta_{PA}, \gamma_{PA}, r'_{PA} \in_R Z_p^*$ and computes $d_{PA,1} = \alpha_{PA}$, $d_{PA,2} = \gamma_{PA}$, $d_{PA,3} = g_1^{\beta+\beta_{PA}\cdot H_1(PID_{PA})}$, $d_{PA,5,i} = (g^a)^{r_{PA}\cdot H_1(attr_i)}$ for $1 \le i \le n_{PA}$. Implicitly set $r_{PA} = b + r'_{PA}$ and calculate

$$d_{PA,4} = g^{\alpha - a\cdot r_{PA}} = g^{(\alpha'+ab)-a\cdot(b+r'_{PA})} = g^{\alpha'-ar'_{PA}} = g^{\alpha'} \cdot (g^a)^{-r'_{PA}}.$$

PA's secret key $SK_{PA} = (d_{PA,1}, d_{PA,2}, d_{PA,3}, d_{PA,4}, \{d_{PA,5,i}\}_{i\in[n_{PA}]})$ is sent to $\mathcal{A}$.

– *Users's secret key query*: $\mathcal{A}$ queries on the user U's attribute secret key with attribute set $S = (attr_1 \cdots attr_{n_U})$. $\mathcal{C}$ computes $PID_U = SEnc(U, H_0(\alpha, \beta))$, selects $\alpha_U, \beta_U, \gamma_U, r'_U \in_R Z_p^*$ and computes $d_{U,1} = \alpha_U$, $d_{U,2} = \gamma_U$, $d_{U,3} = g_1^{\beta+\beta_U\cdot H_1(PID_U)}$, $d_{U,5,i} = (g^a)^{r_U\cdot H_1(attr_i)}$ for $1 \le i \le n_U$. Implicitly set $r_U = b + r'_U$ and calculate

$$d_{U,4} = g^{\alpha - a\cdot r_U} = g^{(\alpha'+ab)-a\cdot(b+r'_U)} = g^{\alpha'-ar'_U} = g^{\alpha'} \cdot (g^a)^{-r'_U}.$$

U's secret key $SK_U = (d_{U,1}, d_{U,2}, d_{U,3}, d_{U,4}, \{d_{U,5,i}\}_{i\in[n_U]})$ is sent to $\mathcal{A}$.

– *Mobile node's secret key query*: $\mathcal{A}$ queries on the mobile node MN's secret key, which belongs to the patient PA's IoT network. Assume that $\mathcal{A}$ has queried PA's secret key SK_{PA}. Then, $\mathcal{C}$ computes $PID_{MN} = SEnc(MN, H_0(SK_{PA}))$, selects $\alpha_{MN} \in_R Z_p^*$ and computes $f_1 = g_1^{\alpha_{PA}+\alpha_{MN}\cdot H_1(PID_{PA}, PID_{MN})}$, $f_2 = g_1^{\frac{1}{\alpha_{PA}-H_1(PID_{MN})}}$ MN's secret key $SK_{MN} = (f_1, f_2)$ is sent to $\mathcal{A}$.

– *Keyword match based policy update query*: $\mathcal{A}$ queries on the patient PA, keyword set $QW = (w_{\sigma_1}, \cdots, w_{\sigma_{l_2}})$ and access policies $(\mathbb{A}, \rho)$, $(\mathbb{A}', \rho')$, $\mathcal{C}$ firstly constructs PA's secret key SK_{PA} as in the "Patient's secret key query". Then, $\mathcal{C}$ selects $s'_i \in_R Z_p^*$ for $1 \le i \le n_1$, and calculates $C'_{2,i} = \rho'(i) \cdot s'_i \cdot \gamma_{PA}^{-1}$ for $1 \le i \le n_1$ and

$q_j = (l_2 \cdot \gamma_{PA})^{-1}(\alpha_{PA})^j \sum_{k=1}^{l_2} H(w_{\sigma_k})^j$ for $0 \leq j \leq n_2$. Then, the policy update query $PUQ = ((\mathbb{A}, \rho), (\mathbb{A}', \rho'), \{C'_{2,i}\}_{i \in [n'_1]}, \{q_j\}_{j \in \{0,\cdots,l_2\}})$ is returned to $\mathcal{A}$.

- *Challenge*: $\mathcal{A}$ sends to $\mathcal{C}$ a challenge access policy $(\mathbb{A}^*, \rho^*)$, a challenge patient identity PA^*, a challenge EHR M^*, two EHR encryption keys $(\Upsilon_0^*, \Upsilon_1^*)$, and two challenge keyword sets (KW_0^*, KW_1^*), where $\mathbb{A}^* \in Z_p^{n_1^* \times n_2^*}$ and ρ^* maps $\mathbb{A}^*$'s rows to attributes. It is required that the secret key of the attribute set S that satisfies the challenge access policy $(\mathbb{A}^*, \rho^*)$ is not queried in phase 1. Moreover, the secret key of PA^* is not queried. The challenge EHR ciphertext is constructed below.

  $\mathcal{C}$ picks $r^*, \gamma_{PA^*} \in_R Z_p^*$ and $s_i^* \in_R Z_p^*$ for $1 \leq i \leq n_1^*$. $\mathcal{C}$ flips random coins $\mu_1, \mu_2 \in \{0, 1\}$ and constructs

$$\begin{aligned} C_M^* &= SEnc(M^*, H_0(\Upsilon_{\mu_1}^*)), \quad C_0^* = \Upsilon_{\mu_1}^* \cdot Z \cdot e(g^{\alpha'}, g^s), \\ C_1^* &= g^s, \quad C_4^* = \theta^r, \\ C_{2,i}^* &= \rho(i)^* \cdot s_i^* \cdot (\gamma_{PA}^*)^{-1}, \ 1 \leq i \leq n_1^*, \end{aligned}$$

  Suppose the challenge keyword set $KW_{\mu_2}^* = (w_1^*, \cdots, w_{l_1}^*)$. $\mathcal{C}$ selects a polynomial $\Gamma^*(x) = a_{l_1}^* x^{l_1} + a_{l_1-1}^* x^{l_1-1} + \cdots + a_1^* x + a_0^*$ such that $\alpha_{PA^*} H(w_1^*), \cdots, \alpha_{PA^*} H(w_{l_1}^*)$ are the l_1 roots of the equation $\Gamma^*(x) = 1$. $\mathcal{C}$ calculates $C_{3,j^*} = r^* \cdot a_j^* \cdot \beta_{PA^*}$ for $0 \leq i \leq l_1$. The challenge EHR ciphertext $CT^* = ((\mathbb{A}^*, \rho^*), C_M^*, C_0^*, C_1^*, \{C_{2,i}^*\}_{i \in [n_1]}, \{C_{3,j}^*\}_{j \in \{0,\cdots,l_1\}}, C_4^*)$ is sent to $\mathcal{A}$.

  It is obvious that if $Z = e(g, g)^{abs}$, then

$$\begin{aligned} C_0^* &= \Upsilon_{\mu_1}^* \cdot Z \cdot e(g^{\alpha'}, g^s) \\ &= \Upsilon_{\mu_1}^* \cdot e(g, g)^{abs} \cdot e(g^{\alpha'}, g^s) \\ &= \Upsilon_{\mu_1}^* \cdot e(g^{ab+\alpha'}, g^s) \\ &= \Upsilon_{\mu_1}^* \cdot e(g, g)^{\alpha s}. \end{aligned}$$

- *Phase 2*: It is the same as in Phase 1 except that the secret key of the attribute set S that satisfies the challenge access policy $(\mathbb{A}^*, \rho^*)$ is not allowed to be queried. In addition, the secret key of PA^* is not allowed to be queried.

- *Guess*: Finally, $\mathcal{A}$ outputs $\mu_1', \mu_2' \in \{0,1\}$. If $\mu_1' = \mu_1$ and $\mu_2' = \mu_2$, $\mathcal{A}$ wins the game. Then, $\mathcal{C}$ could solve the DBDH problem by distinguishing $Z = e(g,g)^{abs}$ or Z is a random element in G_T.

7. Security Requirements Analysis

This IoT and big data fusion system achieves the security requirements defined in Section 3.3.

7.1. Patient anonymity and traceability

When a patient PA registers to the system in Subsection 4.2, an anonymous identity $PID_{PA} = SEnc(PA, H_0(\alpha, \beta))$ is generated by ta, where α, β are the elements in the master secret key MSK. Since MSK is kept secret by ta, no adversary can recover patient's real identity PA from the anonymous identity PID_{PA} due to the security of the symmetric encryption $SEnc$ algorithm. If a patient is found misbehaving, TA can trace his real identity by calculating $PA = SDec(PID_{PA}, H_0(\alpha, \beta))$. Thus, the patient anonymity and traceability are guaranteed.

7.2. User anonymity and traceability

The analysis of user anonymity and traceability is similar to that in Subsection 7.1, which is omitted here.

7.3. Medical node anonymity and traceability

When a medical node MN in patient PA's health IoT network registers to the system in Subsection 4.4, an anonymous identity PID_{MN} is generated by PA through calculating $PID_{MN} = SEnc(MN, H_0(SK_{PA}))$, where SK_{PA} is PA's secret key. Since SK_{PA} is kept secret by PA, no adversary can recover medical nodes's real identity MN from the anonymous identity PID_{MN}

due to the security of the symmetric encryption $SEnc$ algorithm. If a medical node is found misbehaving, PA can trace its real identity by calculating $MN = SDec(PID_{MN}, H_0(SK_{PA}))$. Thus, the medical node anonymity and traceability are guaranteed.

7.4. Confidentiality of IoT key

In the e-health IoT key generation algorithm (shown in Subsection 4.5), the IoT key is constructed as $k = H_0[e(g, g_1)^{r_k}]$, where $r_k \in_R Z_p^*$. The key extraction auxiliary message KEA is constructed as $KEA = (b_1, b_2, b_3, b_4, TS_{PA})$, where TS_{PA} is a time stamp, $b_1 = w_1^{r_k}$, $b_2 = g^{r_k \prod_{j=1}^{n} H_1(PID_{MN_j})}$, $b_3 = g^{\alpha_k}$ and $b_4 = d_3 \cdot g_1^{\alpha_k \cdot H_1(PA, \Sigma_{MN}, b_1, b_2, b_3, k, TS_{PA})}$. The adversary cannot recover the random number r_k from b_1 or b_2 due to the intractability of discrete logarithmic problem (DLP). Then, the adversary cannot deduce the IoT key k from the key extraction auxiliary message KEA. Thus, the confidentiality of IoT key is ensured.

7.5. Authentication of IoT key distribution

In the authenticated e-health IoT key extraction algorithm (shown in Subsection 4.6), the medical node MN in patient PA's IoT network utilizes the element $f_2 = g_1^{\frac{1}{\alpha_{PA} - H_1(PID_{MN})}}$ in its secret key SK_{MN} to recover the IoT key $k = H_0[e(b_1 \cdot b_2^{1/\prod_{j=1, j\neq i}^{n} H_1(PID_{MN_j})}, f_2)]$. Then, MN authenticates that whether the received key extraction auxiliary message KEA is sent from PA by calculating

$$e(g, b_4) = e(g_1, g_2 \cdot (w_{PA,2})^{H_1(PID_{PA})} \cdot b_3^{H_1(PA, \Sigma_{MN}, b_1, b_2, b_3, k, TS_{PA})}).$$

KEA is authenticated to be a valid key extraction auxiliary message from PA when the above equation holds.

Since $b_4 = d_{PA,3} \cdot g_1^{\alpha_k \cdot H_1(PA, \Sigma_{MN}, b_1, b_2, b_3, k, TS_{PA})}$, $d_{PA,3} = g_1^{\beta + \beta_{PA} \cdot H_1(PID_{PA})}$, we have $b_4 = g_1^{\beta + \beta_{PA} \cdot H_1(PID_{PA}) + \alpha_k \cdot H_1(PA, \Sigma_{MN}, b_1, b_2, b_3, k, TS_{PA})}$. A valid element b_4 cannot be constructed without β_{PA}, which is an element in PA's secret key SK_{PA}. In the above verification equation, an element $w_{PA,2}$ of PA's public key PK_{PA} is utilized to verify whether b_4 contains the element β_{PA}. Thus, a valid KEA cannot be generated without PA's secret key SK_{PA} and the authentication of IoT key distribution is realized.

7.6. Confidentiality of IoT message

In the IoT message encryption algorithm (shown in Subsection 4.7), the ciphertext of IoT message is $C_m = (\Phi_{m,0}, \Phi_{m,1}, \Phi_{m,2}, TS_{MN})$, where $\Phi_{m,0} = SEnc(m, k)$, $\Phi_{m,1} = g^{r_m}$, $\Phi_{m,2} = f_1 \cdot g_1^{r_m \cdot H_1(PID_{PA}, PID_{MN}, \Phi_{m,0}, TS_{MN})}$ and TS_{MN} is a time stamp. Since the IoT message m is encrypted to $\Phi_{m,0} = SEnc(m, k)$, the confidentiality of IoT message is guaranteed by the confidentiality of the IoT key k and the security of $SEnc$ algorithm.

7.7. Authentication of IoT message

In the e-health IoT message verification and decryption algorithm (shown in Subsection 4.8), the source of IoT ciphertext is authenticated by the equation

$$\begin{aligned} & e(g, \Phi_{m,2}) \\ = \; & e[g_1, w_{PA,1} \cdot (PK_{MN})^{H_1(PID_{PA}, PID_{MN})} \cdot (\Phi_{m,1})^{H_1(PID_{PA}, PID_{MN}, \Phi_{m,0}, TS_{MN})}]. \end{aligned}$$

Since the IoT ciphertext element $\Phi_{m,2} = f_1 \cdot g_1^{r_m \cdot H_1(PID_{PA}, PID_{MN}, \Phi_{m,0}, TS_{MN})}$ and $f_1 = g_1^{\alpha_{PA} + \alpha_{MN} \cdot H_1(PID_{PA}, PID_{MN})}$ is an element in MN's secret key SK_{MN}, $\Phi_{m,2}$ cannot be constructed without f_1. In the above verification equation, MN's public key $PK_{MN} = g^{\alpha_{MN}}$ is utilized to verify whether $\Phi_{m,2}$ contains the element α_{MN}. Thus, a valid IoT ciphertext C_m cannot be generated without MN's secret key SK_{MN} and the authentication of IoT message is realized.

7.8. Secure against replay attack

In the e-health IoT key distribution procedure (includes e-health IoT key generation algorithm and authenticated e-health IoT key extraction algorithm), a time stamp TS_{PA} is selected by the patient PA to indicate the IoT key generation time. The time stamp TS_{PA} is embedded in the key extraction auxiliary message KEA by calculating $b_4 = d_{PA,3} \cdot g_1^{\alpha_k \cdot H_1(PA, \Sigma_{MN}, b_1, b_2, b_3, k, TS_{PA})}$, and verified in the authentication equation $e(g, b_4) = e(g_1, g_2 \cdot (w_{PA,2})^{H_1(PID_{PA})} \cdot b_3^{H_1(PA,\Sigma_{MN},b_1,b_2,b_3,k,TS_{PA})})$. If the adversary replays the key extraction auxiliary message KEA utilizing the original time stamp, the medical node MN will discover the replay attack by checking the history time stamps. If the adversary replays the key extraction auxiliary message KEA utilizing a forged time stamp, the authentication equation will not hold and the replay attack will be discovered too.

In the e-health IoT message encryption algorithm, a time stamp TS_{MN} is selected by the medical node MN to indicate the encryption time. The time stamp TS_{MN} is embedded in the IoT ciphertext by calculating $\Phi_{m,2} = f_1 \cdot g_1^{r_m \cdot H_1(PID_{PA},PID_{MN},\Phi_{m,0},TS_{MN})}$, and verified in the authentication equation

$$\begin{aligned} & e(g, \Phi_{m,2}) \\ = \; & e[g_1, w_{PA,1} \cdot (PK_{MN})^{H_1(PID_{PA},PID_{MN})} \cdot (\Phi_{m,1})^{H_1(PID_{PA},PID_{MN},\Phi_{m,0},TS_{MN})}]. \end{aligned}$$

If the adversary replays the IoT ciphertext C_m utilizing the original time stamp, the patient PA will discover the replay attack by checking the history time stamps. If the adversary replays the IoT ciphertext C_m utilizing a forged time stamp, the authentication equation will not hold and the replay attack will be discovered too.

Thus, this system is secure against replay attack.

8. Performance Analysis and Comparison

In this section, we firstly compare the proposed fusion of IoT and big data system with the state-of-the art schemes [31, 32, 33, 34, 35, 36] that support fine-grained access control mechanism. Then, they are evaluated using the simulation in terms of communication and computation costs.

8.1. Comparison

The function, communication and computation overheads of the schemes in [31, 32, 33, 34, 35, 36] and our fusion of IoT and big data system are compared. The notations in Table 2 are used in the comparison.

Table 2: The Performance Notations

Notation	Description
e_1	Exponential operation in G
e_2	Exponential operation in G_T
e_p	Elliptic curve bilinear pairing operation
$\|G\|$	The size of element in group G
$\|G_T\|$	The size of element in group G_T
$\|Z_p\|$	The size of element in Z_p
$\|Att\|$	The number of total attributes in the system
$\|S\|$	The number of attributes in attribute set S
n_1	The row number of matrix $\mathbb{A}$
n_1'	The row number of matrix $\mathbb{A}'p$
l_1	The number of the keywords extracted from the file
l_2	The number of the query keywords
τ	The number of IoT messages in the batch verification

8.1.1. Function Comparison

The function of the schemes in [31, 32, 33, 34, 35, 36] and our fusion of e-health IoT and big data scheme is compared in Table 3.

- ***Anonymity and traceability***: The proposed system realizes the anonymity and traceability of patient and medical nodes. The other schemes [31, 32, 33, 34, 35, 36] do not take this important issue into consideration.

- ***IoT message encryption***: The medical data collected by the medical nodes are encrypted using the IoT key and then transmitted to the patient in our system, which protects the confidentiality of the patient's vital signs. This function is not supported by the other schemes [31, 32, 33, 34, 35, 36].

- ***IoT message authentication***: The source authentication of IoT message is realized our system, which effectively prevents the impersonation attack. The other schemes [31, 32, 33, 34, 35, 36] do not realize this function.

- ***Authenticated IoT key distribution***: In the IoT key distribution phase, the medical nodes are capable to authenticate whether the IoT key is sent by the patient. The scheme in [36] and our system are able to realize this function, while the other schemes [31, 32, 33, 34, 35] cannot support authenticated IoT key distribution.

- ***Policy update***: The schemes in [31, 32, 33] and our system realize dynamic policy updating for the outsourced stored data, while the other schemes [35, 36] do not realize this function.

- ***Keyword match based update***: In the medical big data system, the patients may have a large amount of encrypted EHRs that are encrypted using the same access policy. This proposed system enables the patient to update only part of these EHRs using the keyword match mechanism, while all the other schemes [31, 32, 33, 34, 35] do not consider this problem.

- ***Keyword match***: The keyword match function is studied in the [34, 35] to construction searchable encryption schemes. Our system utilizes the keyword match mechanism to control the dynamic access policy update. The other schemes [31, 32, 33] do not support keyword match function.

- ***Flexible subset match***: The scheme in [34] only realizes single keyword search and that in [35] supports conjunctive keyword search. Flexible subset match function is realized in our system such that the encrypted EHR is deemed as match file if the query keyword set is a subset of the pre-defined keyword set in the EHR.

8.1.2. Communication Overhead Comparison

Table 4 shows the communication overhead of our secure fusion system. The sizes of system public parameter PP and master secret key MSK are constants. Thus, our scheme is a large universe construction and supports arbitrary number of attributes. The patient's (or data user) public key PK_{PA} (or PK_U) consists of three elements in group G and secret key SK_{PA} (or SK_U) has size linearly grow with the attribute size $|S|$. The medical node's public/secret keys PK_{MN}/SK_{MN} consist of one element and two element in group G, respectively. The size of IoT key extraction auxiliary message KEA is $4|G|$ and that of C_m is $2|G|$. The size of CT linearly grow with n_1 and l_1, and the size of the policy update query PUQ linearly grow with n_1' and l_2.

In Table 5, the sizes of public parameter, user's (or patient) attribute secret key and (EHR) file ciphertext are compared. (1) The schemes in [31, 32, 33, 35, 36] and our system all have constant public parameter size. However, $|PP|$ of the scheme [34] depends on the total attribute size $|Attr|$ in the system. (2) The sizes of attribute secret key in schemes [31, 32, 33, 34, 35, 36] and our system increase with $|S|$. The schemes in [32, 34, 35, 36] has $|SK|$ larger than our system. (3) The size of (EHR) file ciphertext schemes [31, 32, 33, 34, 35, 36] and our system increases on the row number n_1 of the matrix $\mathbb{A}$ in the access policy. CT in our system has $(n_1 + l_1)$ elements in Z_p, two elements in G and

Table 3: Function Comparison

Scheme	**[31]**	**[32]**	**[33]**	**[34]**	**[35]**	**[36]**	**Ours**
F1	×	×	×	×	×	×	√
F2	×	×	×	×	×	×	√
F3	×	×	×	×	×	×	√
F4	×	×	×	×	×	×	√
F5	×	×	×	×	×	×	√
F6	×	×	×	×	×	√	√
F7	√	√	√	×	×	×	√
F8	×	×	×	×	×	×	√
F9	×	×	×	√	√	×	√
F10	×	×	×	×	×	×	√

$F1$: patient anonymity

$F2$: patient traceability

$F3$: medical node anonymity and traceability

$F4$: IoT message encryption

$F5$: IoT message authentication

$F6$: authenticated IoT key distribution

$F7$: policy update

$F8$: keyword match based update

$F9$: keyword match

$F10$: flexible subset match

Table 4: Communication Overhead of Our System

Parameter	**Communication Overhead**
$\|PP\|$	$3\|G\| + \|G_T\|$
$\|MSK\|$	$3\|Z_p\|$
$\|PK_{PA}\|$	$3\|G\|$
$\|SK_{PA}\|$	$2\|Z_p\| + (\|S\| + 2)\|G\|$
$\|PK_U\|$	$3\|G\|$
$\|SK_U\|$	$2\|Z_p\| + (\|S\| + 2)\|G\|$
$\|PK_{MN}\|$	$\|G\|$
$\|SK_{MN}\|$	$2\|G\|$
$\|KEA\|$	$4\|G\|$
$\|C_m\|$	$2\|G\|$
$\|CT\|$	$(n_1 + l_1)\|Z_p\| + 2\|G\| + \|G_T\|$
$\|PUQ\|$	$(n'_1 + l_2 + 1)\|Z_p\|$

one elements in G_T. CT in scheme [35] linearly grow with l_1 and $|Attr|$, which is larger than the other schemes [31, 32, 33, 34, 36] and our system.

In Table 6, the size of access policy update query PUQ in our system is compared that in schemes [31, 32, 33] and the size of keyword index is compared with that in [34, 35]. (1) The schemes [31, 32, 33] have similar policy update mechanism, where the new access policy is compared with the previous policy. The attributes in the new access policy are classified into three types, and the numbers of the the three types of attributes are denoted as $n'_{11}, n'_{12}, n'_{13}$ such that $n'_{11} + n'_{12} + n'_{13} = n'_1$, where n'_1 is the row number of $\mathbb{A}$. In order to simplify the comparison, we set $n'_{11} = n'_{12} = n'_{13} = n'_1/3$. Since the element bit length in Z_p is less than that in G, the size of access policy update query in our system is less than that in [31, 32, 33]. (2) The sizes of keyword index in schemes [34, 35] are $(2l_1 + 2|S| + 1)|G|$ and $(l_1 + 2|S| + 2)|G|$, respectively. That in in our system is $(l_1)|Z_p| + |G|$, which is less that the other schemes [34, 35].

Table 5: Communication Overhead Comparison 1

Scheme	**Z1**	**Z2**	**Z3**
KanY [31]	$\lvert G\rvert$	$(\lvert S\rvert)\cdot\lvert G\rvert$	$(2n_1)\lvert G\rvert$ $+(n_1+1)\lvert G_T\rvert$
Hongwei [32]	$2\lvert G\rvert+2\lvert G_T\rvert$	$(2\lvert S\rvert)\cdot\lvert G\rvert$	$(2n_1+1)\lvert G\rvert$ $+(n_1+1)\lvert G_T\rvert$
Ying [33]	$2\lvert G\rvert$	$(\lvert S\rvert)\cdot\lvert G\rvert$	$(2n_1+1)\lvert G\rvert$ $+\lvert G_T\rvert$
Jiguo [34]	$(\lvert Att\rvert+4)\lvert G\rvert$	$(2\lvert S\rvert)\lvert G\rvert$	$(n_1+2)\lvert G\rvert$ $+\lvert G_T\rvert$
Miao [35]	$5\lvert G\rvert+\lvert G_T\rvert$	$(2\lvert S\rvert+2)\lvert G\rvert$	$(n_1+2\lvert Att\rvert+2)\lvert G\rvert$
Yeh [36]	$\lvert G\rvert+\lvert G_T\rvert$	$(2\lvert S\rvert+1)\lvert G\rvert$	$(2n_1+1)\lvert G\rvert$ $+\lvert G_T\rvert$
Ours	$3\lvert G\rvert+\lvert G_T\rvert$	$2\lvert Z_p\rvert$ $+(\lvert S\rvert+2)\lvert G\rvert$	$(n_1+l_1)\lvert Z_p\rvert$ $+2\lvert G\rvert+\lvert G_T\rvert$

$Z1$: size of public parameter $|PP|$

$Z2$: size of user's (or patient) attribute secret key $|SK_U|$ (or $|SK_{PA}|$)

$Z3$: size of (EHR) file ciphertext $|CT|$

Table 6: Communication Overhead Comparison 2

Scheme	**Z4**	**Z5**
KanY [31]	$(2n'_1 + n'_{13})\|G\|$	$\perp$
Hongwei [32]	$(2n'_1 + n'_{13})\|G\|$	$\perp$
Ying [33]	$(n'_1 + n'_{13})\|G\|$	$\perp$
Jiguo [34]	$\perp$	$(2l_1 + 2\|S\| + 1)\|G\|$
Miao [35]	$\perp$	$(l_1 + 2\|S\| + 2)\|G\|$
Ours	$(n'_1 + l_2 + 1)\|Z_p\|$	$(l_1)\|Z_p\| + \|G\|$

$Z4$: size of access policy update query $|PUQ|$

$Z5$: size of keyword index

8.1.3. Computation Overhead Comparison

Table 7 shows the communication overhead of our system. The global setup algorithm requires one exponentiation operation in G, one exponentiation operation in G_T and one bilinear pairing calculation. The computation overheads of patient, user and medical node registration algorithms are $(|S|+5)e_1$, $(|S|+5)e_1$ and $3e_1$, respectively. The IoT key generation algorithm consumes 4 exponentiation operations in G, and IoT key extraction algorithm needs 3 exponentiation operations in G and 3 bilinear pairing calculation. The computation overheads of IoT message encryption algorithm, IoT message verification and decryption algorithm, and IoT message batch verification and decryption algorithm are $2e_1$, $2e_1 + 2e_p$ and $(3\tau)e_1 + 2e_p$, respectively, where τ is the number of IoT messages in the batch verification. The EHR file encryption algorithm consumes two exponentiation operation in G and one exponentiation operation in G_T. The policy update query generation algorithm does not require any exponentiation or bilinear paring calculation, which is denoted as "≈ 0". The computation overheads of ciphertext update algorithm and EHR decryption algorithms are e_2 and $(n_1)e_1 + e_p$, respectively.

In Table 8, our system is compared with the schemes in [31, 32, 33, 34, 35, 36].

Table 7: Computation Overhead of Our System

Algorithm	**Computation Overhead**
Alg. 1: Global Setup	$e_1 + e_2 + e_p$
Alg. 2: Patient Reg.	$(\|S\| + 5)e_1$
Alg. 3: User Reg.	$(\|S\| + 5)e_1$
Alg. 4: Medical Node Reg.	$3e_1$
Alg. 5: IoT Key Gen.	$4e_1$
Alg. 6: IoT Key Extraction	$3e_1 + 3e_p$
Alg. 7: IoT Msg. Enc.	$2e_1$
Alg. 8: IoT Msg. Ver. & Dec	$2e_1 + 2e_p$
Alg. 9: IoT Bat. Ver. & Dec	$(3\tau)e_1 + 2e_p$
Alg. 10: EHR Enc.	$2e_1 + e_2$
Alg. 12: Policy Update Query	≈ 0
Alg. 12: Ciphertext Update	e_2
Alg. 11: Dec	$(n_1)e_1 + e_p$

(1) The computation costs for user (or patient) attribute secret key of these schemes linearly increase with $|S|$. Our system requires $|S|+2$ exponentiation operations in group G to generate an attribute secret key, which is smaller than that in the other schemes [31, 32, 33, 34, 35, 36]. (2) The computation costs for (EHR) file encryption in schemes [31, 32, 33, 34, 35, 36] linearly grow with the parameter n_1, which is the row number of matrix $\mathbb{A}$ in the access policy. Our system requires two exponentiation operations in group G and one exponentiation operations in group G_T to generate a EHR ciphertext, which is much smaller than that in the other schemes [31, 32, 33, 34, 35, 36]. (3) The computation costs for (EHR) file decryption also depends on n_1. Our system requires n_1 exponentiation operations in group G and two bilinear pairing operations to decrypt a EHR ciphertext, which is smaller than that in [31, 32, 33, 35, 36] and larger than that in [34].

Table 9 shows the computation cost for access policy update query generation algorithm and ciphertext policy update algorithm. (1) The schemes in [31, 32, 33] require $(2n'_1 + 3n'_{13})e_1$, $(2n'_1 + 3n'_{13})e_1$ and $(n'_1 + 2n'_{13})e_1$ to generate PUQ, while our system does not need any exponentiation or bilinear pairing calculation. (2) In the ciphertext policy update phase, our system consumes one exponentiation operation in group G_T, which is much smaller than the computation costs in other schemes [31, 32, 33].

Table 10 shows the computation cost for keyword index generation and keyword match test. (1) The schemes in [34, 35] require e_1 and $(2l_2)e_p$ to generate encrypted keyword index, while the computation cost of that in our system is one exponentiation operation in group G. (2) In the keyword match test phase, our system consumes one exponentiation operation in group G_T, which is much smaller than the computation costs in other schemes [34, 35].

8.2. Simulation

The performance of this proposed system and the other schemes in [31, 32, 33, 34, 35, 36] are evaluated by the simulations, which are implemented on a personal computer running Windows 10 and 64-bit operation system with

Table 8: Computation Overhead Comparison 1

Scheme	**T1**	**T2**	**T3**
KanY [31]	$(2\|S\|)e_1$	$(2n_1)e_1 + e_p$ $+(2n_1+1)e_2$	$(3n_1)e_2$ $+(3n_1)e_p$
Hongwei [32]	$(3\|S\|)e_1$	$(3n_1+1)e_1$ $+(2n_1+1)e_2 + e_p$	$(2n_1)e_1 + (n_1)e_2$ $+(3n_1)e_p$
Ying [33]	$(\|S\|)e_1$ $+(2\|S\|)e_p$	$(3n_1+1)e_1$ $+(2n_1+1)e_2 + e_p$	$(n_1)e_2$ $+(2n_1+1)e_p$
Jiguo [34]	$(3\|S\|)e_1$	$(n_1+2)e_1$ $+e_2$	$2e_1 + 3e_p$
Miao [35]	$(3\|S\|+2)e_1$	$(4n_1)e_1$	$(n_1)e_2 + (n_1)e_p$
Yeh [36]	$(3\|S\|+1)e_1$	$(2n_1+1)e_1$ $+e_2$	$(n_1)e_1 + (2n_1)e_p$
Ours	$(\|S\|+2)e_1$	$2e_1 + e_2$	$n_1 \cdot e_1 + 2e_p$

$T1$: computation cost for user (or patient) attribute secret key

$T2$: computation cost for (EHR) file encryption

$T3$: computation cost for (EHR) file decryption

Table 9: Computation Overhead Comparison 2

Scheme	**T4**	**T5**
KanY [31]	$(2n'_1 + 3n'_{13})e_1$	$(n'_{13})e_p$
Hongwei [32]	$(2n'_1 + 3n'_{13})e_1$	$(2n'_{12})e_1 + (n'_{12})e_2 + (n'_1)e_p$
Ying [33]	$(n'_1 + 2n'_{13})e_1$	$(2n'_1 + n'_{12})e_1$
Ours	≈ 0	e_2

$T4$: computation cost for access policy update query generation

$T5$: computation cost for ciphertext policy update

Table 10: Computation Overhead Comparison 3

Scheme	**T6**	**T7**
Jiguo [34]	e_1	$(2l_2)e_p$
Miao [35]	$(3l_1+2\|S\|+1)e_1$	$(n_1)e_2+(2n_1+3l_2)e_p$
Ours	e_1	e_2

$T6$: computation cost for keyword index generation

$T7$: computation cost for keyword match test

Intel(R) Core(TM) i3-2120 CPU @ 3.30 GHz, 4.00 GB RAM. The Type-A elliptic curve $E/F_p: \ y^2 = x^3 + x$ over F_p is selected to conduct experiments, and the group G has order q, where the bit length of p and q are 160-bit and 512-bit, respectively. Thus, the lengths of the elements in G, G_T and Z_p are 1024-bit, 1024-bit and 160-bit, respectively.

8.2.1. Transmission Efficiency

Fig. 6 shows the transmission cost of public parameter and the value of x-axis varies with the number of total attributes. The sizes of public parameter in schemes [31, 32, 33, 35, 36] and our system are constants, which are 0.512 KB, 0.128 KB, 0.512 KB, 0.256 KB, 0.768 KB and 0.256 KB, respectively. $|PP|$ in [34] grows with $|Att|$, which is 3.072 KB when $|Att| = 20$ and 13.312 KB when $|Att| = 100$.

Fig. 7 shows the transmission cost of user's attribute secret key and the value of x-axis varies with the number of attributes. When $|S| = 20$, the sizes of user's attribute secret key in schemes [31, 32, 33, 34, 35, 36] and our system are 2.56 KB, 5.12 KB, 2.56 KB, 5.12 KB, 5.376 KB, 5.248 KB and 2.856 KB , respectively. When $|S| = 100$, the sizes of user's attribute secret key in schemes [31, 32, 33, 34, 35, 36] and our system are 12.8 KB, 25.6 KB, 12.8 KB, 25.6 KB, 25.856 KB, 25.728 KB and 13.096 KB , respectively. Thus, our system has good efficiency in transmitting user's attribute secret key.

Fig. 8 shows the transmission cost of CT and the value of x-axis varies

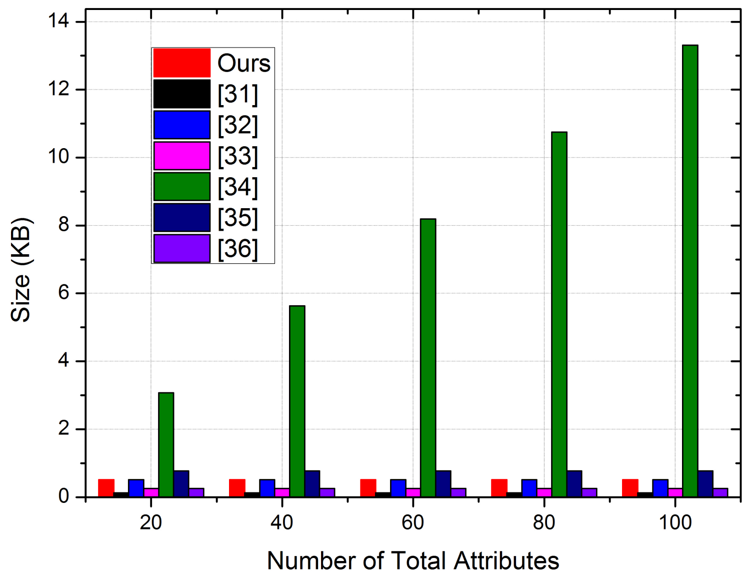

Figure 6: Transmission Cost of Public Parameter

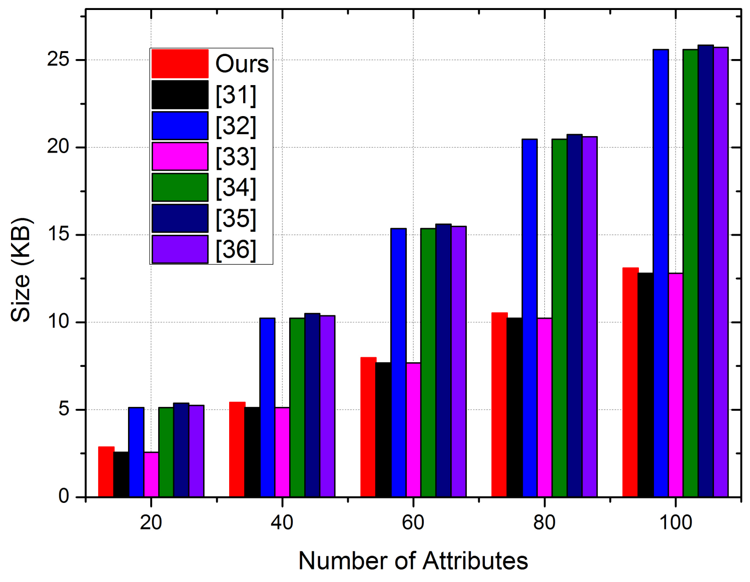

Figure 7: Transmission Cost of User's Attribute Secret Key

with the number of attributes. When $|n_1| = 20$, the sizes of CT in schemes [31, 32, 33, 34, 35, 36] and our system are 7.808 KB, 7.936 KB, 5.376 KB, 2.944 KB, 28.416 KB, 5.376 KB and 1.012 KB , respectively. When $|n_1| = 100$, the sizes of user's attribute secret key in schemes [31, 32, 33, 34, 35, 36] and our system are 38.528 KB, 38.656 KB, 25.856 KB, 13.184 KB, 38.656 KB, 25.856 KB and 2.612 KB , respectively. It is obvious that our system has the least $|CT|$ transmission cost.

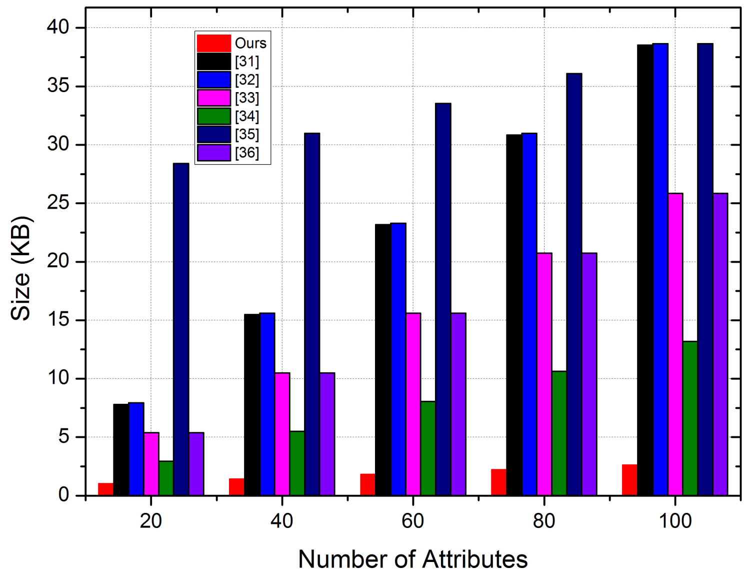

Figure 8: Transmission Cost of Ciphertext

Fig. 9 shows the transmission cost of PUQ and the value of x-axis varies with the number of attributes. When $|n'_1| = 15$, the sizes of PUQ in schemes [31, 32, 33] and our system are 4.48 KB, 4.48 KB, 2.56 KB and 0.42 KB , respectively. When $|n'_1| = 60$, the sizes of user's attribute secret key in schemes [31, 32, 33] and our system are 17.92 KB, 17.92 KB, 10.24 KB and 1.32 KB , respectively. Our system also has the least $|PUQ|$ transmission cost.

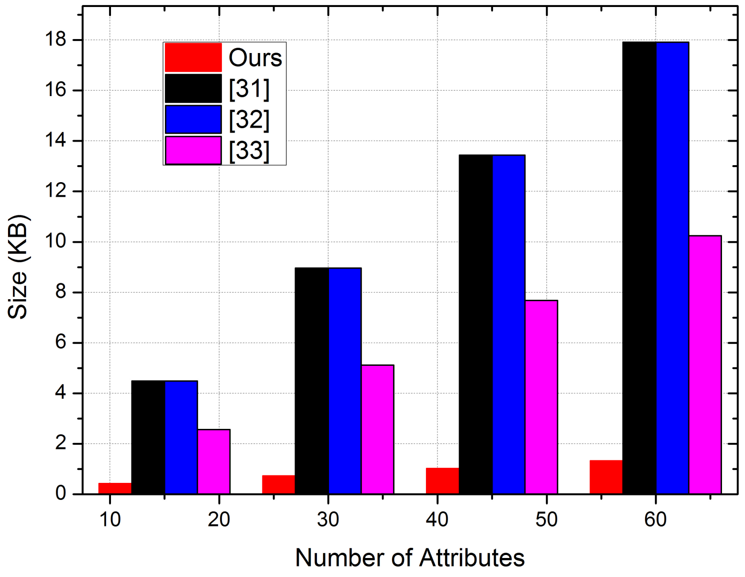

Figure 9: Transmission Cost of Policy Update Query

Fig. 10 shows the transmission cost of keyword encrypted index and the value of x-axis varies with the number of extracted keywords from the (EHR) file. When $|l_1| = 5$, the sizes of keyword encrypted index in schemes [34, 35] and our system are 6.528 KB, 6.016 KB and 0.228 KB , respectively. When $|l_1| = 30$, the sizes of keyword encrypted index in schemes [34, 35] and our system are 12.928 KB, 9.216 KB and 0.728 KB , respectively. Our system has the least keyword encrypted index transmission cost.

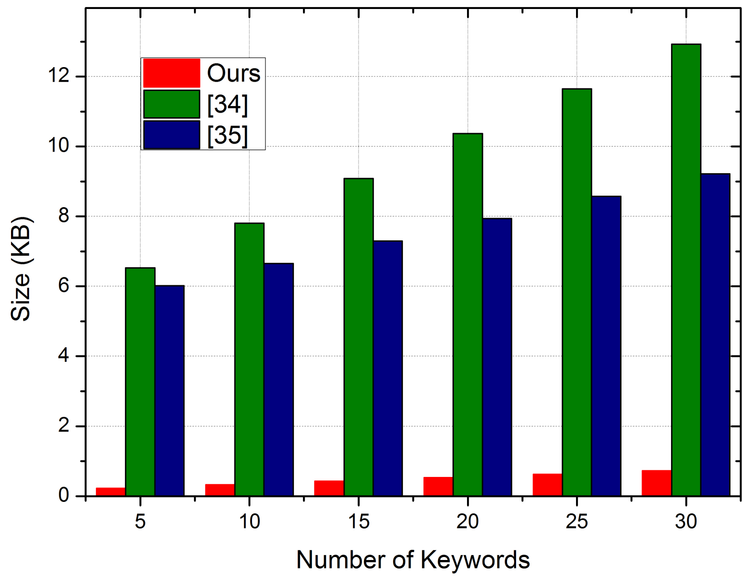

Figure 10: Transmission Cost of Keyword Encrypted Index

8.2.2. Computation Efficiency

Fig. 11 shows the computation cost of user's attribute secret key generation and the value of x-axis varies with the number of user's attributes. When $|S| = 20$, the computation cost in schemes [31, 32, 33, 34, 35, 36] and our system are 0.368 s, 0.552 s, 0.905 s, 0.552 s, 0.57 s, 0.561 s and 0.202 s, respectively. When $|S| = 100$, the computation cost in schemes [31, 32, 33, 34, 35, 36] and our system are 1.839 s, 2.759 s, 4.525 s, 2.759 s, 2.777 s, 2.768 s and 0.938 s, respectively. Our system has the least attribute secret key generation computation cost.

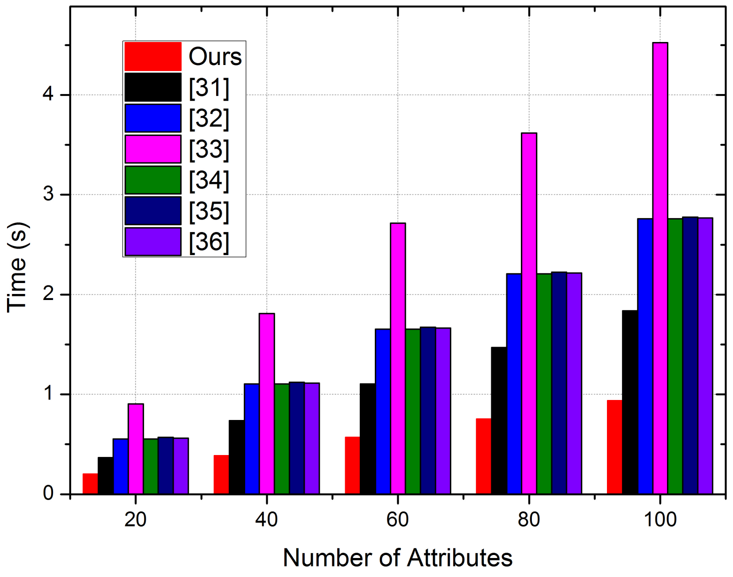

Figure 11: Computation Cost of User's Attribute Secret Key Generation

Fig. 12 and Table 11 show the computation cost of (EHR) file encryption and the value of x-axis varies with n_1. When $|n_1| = 20$, the computation cost in schemes [31, 32, 33, 34, 35, 36] and our system are 0.492 s, 0.689 s, 0.685 s, 0.205 s, 0.736 s, 0.380 s and 0.021 s, respectively. When $|n_1| = 100$, the computation cost in schemes [31, 32, 33, 34, 35, 36] and our system are 2.377 s, 3.299 s, 3.305

s, 0.941 s, 3.678 s, 1.851 s and 0.022 s, respectively. Our system has the least (EHR) file encryption computation cost.

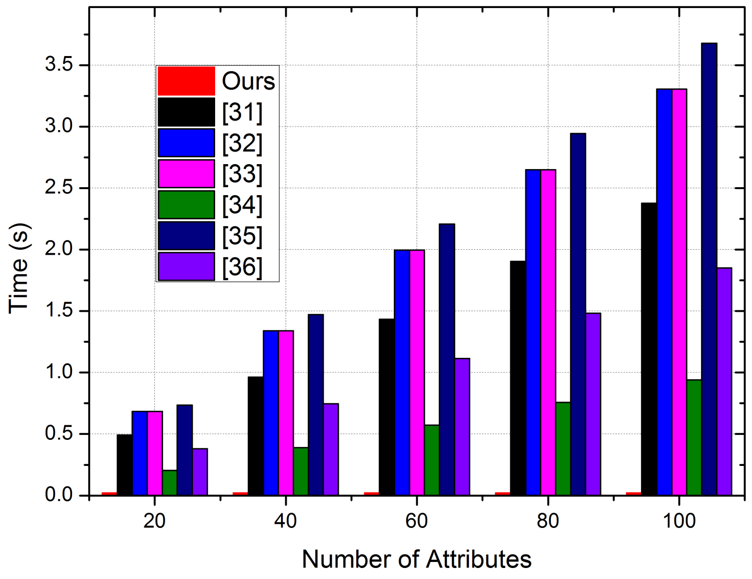

Figure 12: Computation Cost of EHR Encryption

Fig. 13 shows the computation cost of (EHR) file decryption and the value of x-axis varies with n_1. When $|n_1| = 20$, the computation cost in schemes [31, 32, 33, 34, 35, 36] and our system are 1.237 s, 1.501 s, 0.791 s, 0.072 s, 0.412 s, 0.905 s and 0.22 s, respectively. When $|n_1| = 100$, the computation cost in schemes [31, 32, 33, 34, 35, 36] and our system are 6.183 s, 7.506 s, 3.882 s, 0.072 s, 2.061 s, 4.525 s and 0.956 s, respectively. Our system has (EHR) file decryption computation cost lower than that in [31, 32, 33, 35, 36] and higher than that in [34].

Fig. 14 and Table 12 show the computation cost of access policy update query generation and the value of x-axis varies with n_1'. When $|n_1'| = 15$, the computation cost in schemes [31, 32, 33] and our system are 0.419 s, 0.414 s,

Table 11: Computation Cost of EHR Encryption

$\mathbf{n_1}$	**Ours**	[31]	[32]	[33]	[34]	[35]	[36]
20	**0.021**	0.492	0.689	0.685	0.205	0.736	0.380
40	**0.029**	0.963	1.338	1.340	0.389	1.471	0.747
60	**0.025**	1.434	1.988	1.995	0.573	2.207	1.115
80	**0.019**	1.905	2.642	2.650	0.757	2.943	1.483
100	**0.022**	2.377	3.299	3.305	0.941	3.678	1.851

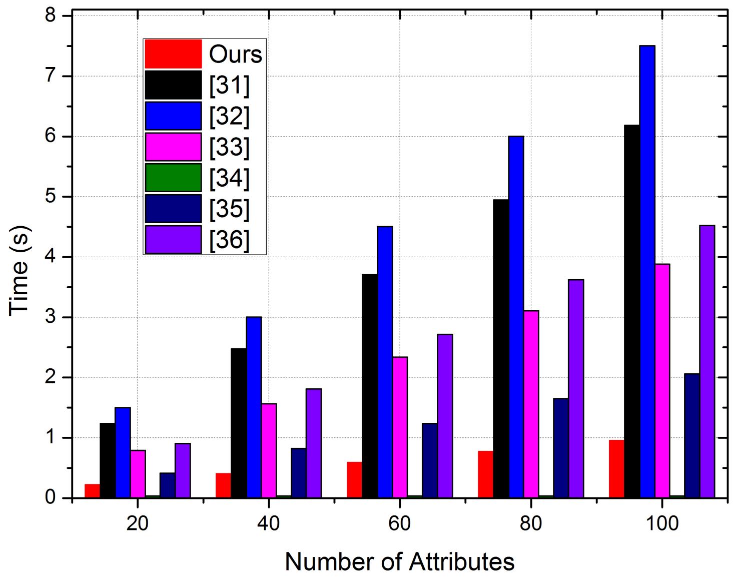

Figure 13: Computation Cost of EHR Decryption

0.237 s and 0.0009 s, respectively. When $|n_1'| = 60$, the computation cost in schemes [31, 32, 33] and our system are 1.658 s, 1.655 s, 0.928 s and 0.0011 s, respectively. Our system has the least access policy update query generation computation cost.

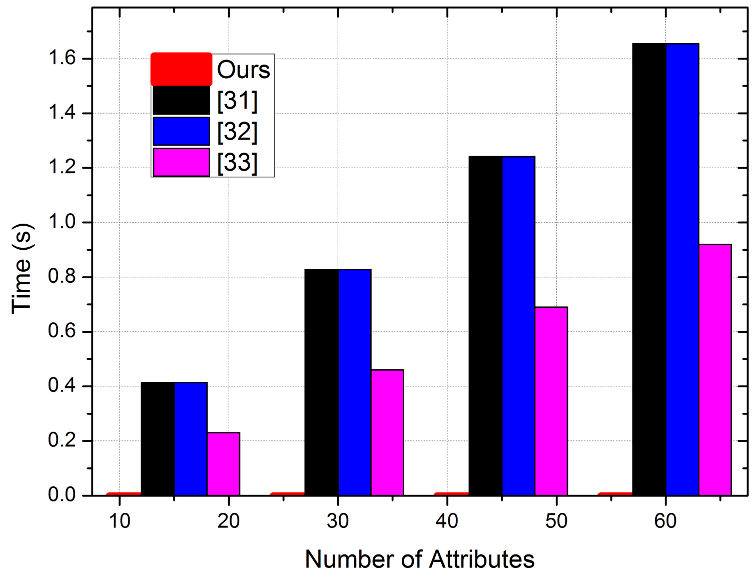

Figure 14: Computation Cost of Access Policy Update Query

Fig. 15 and Table 13 show the computation cost of ciphertext access policy update and the value of x-axis varies with n_1'. When $|n_1'| = 15$, the computation cost in schemes [31, 32, 33] and our system are 0.092 s, 0.375 s, 0.322 s and 0.0029 s, respectively. When $|n_1'| = 60$, the computation cost in schemes [31, 32, 33] and our system are 0.361 s, 1.501 s, 1.287 s and 0.0039 s, respectively. Our system has the least ciphertext access policy update computation cost.

Fig. 16 and Table 14 show the computation cost of keyword index generation and the value of x-axis varies with l_1. To simplify the comparison, the attribute size $|S|$ is set to be 10 in scheme [34]. When $|l_1| = 5$, the computation cost in

Table 12: Computation Cost of Access Policy Update Query

$\mathbf{n}'_1$	**Ours**	[31]	[32]	[33]
15	**0.0009**	0.419	0.414	0.237
30	**0.0009**	0.825	0.828	0.460
45	**0.0014**	1.232	1.241	0.692
60	**0.0011**	1.658	1.655	0.928

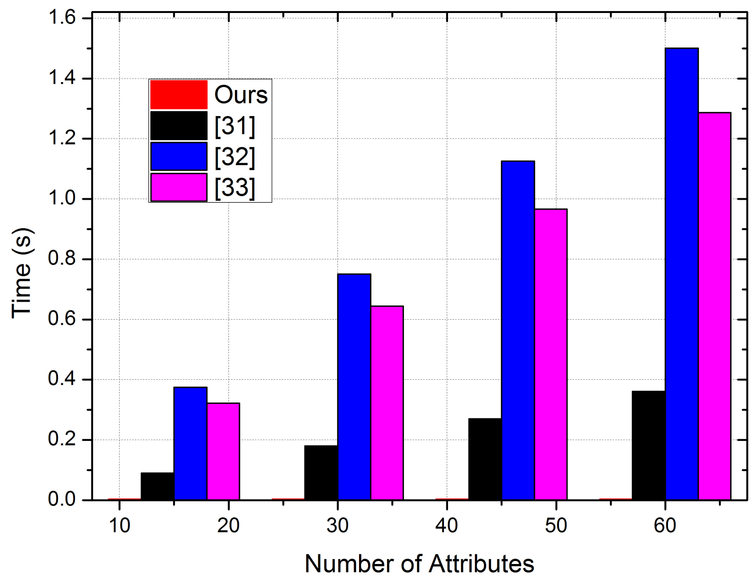

Figure 15: Computation Cost of Ciphertext Update

Table 13: Computation Cost of Ciphertext Update

$\mathbf{n_1'}$	**Ours**	[31]	[32]	[33]
15	**0.0029**	0.092	0.375	0.322
30	**0.0032**	0.186	0.751	0.644
45	**0.0037**	0.275	1.126	0.966
60	**0.0039**	0.361	1.501	1.287

schemes [34, 35] and our system are 0.330 s, 0.0089 s and 0.0091 s, respectively. When $|l_1| = 30$, the computation cost in schemes [34, 35] and our system are 1.018 s, 0.0092 s and 0.0091 s, respectively. Our system has the least keyword index generation computation cost. Although the scheme in [35] is also efficient, it can not realize flexible subset keyword search.

Table 14: Computation Cost of Keyword Index Generation

$\mathbf{n_1'}$	**Ours**	[34]	[35]
5	**0.0091**	0.330	0.0089
10	**0.0093**	0.468	0.0085
15	**0.0088**	0.605	0.0096
20	**0.0096**	0.743	0.0091
25	**0.0092**	0.881	0.0095
30	**0.0091**	1.018	0.0092

Fig. 17 and Table 15 show the computation cost of keyword match test and the value of x-axis varies with l_1. To simplify the comparison, the matrix row number n_1 is set to be 10 in scheme [35]. When $|l_1| = 5$, the computation cost in schemes [34, 35] and our system are 0.180 s, 0.657 s and 0.0031 s, respectively. When $|l_1| = 30$, the computation cost in schemes [34, 35] and our system are 1.082 s, 2.009 s and 0.0034 s, respectively. Our system has the least keyword match test cost.

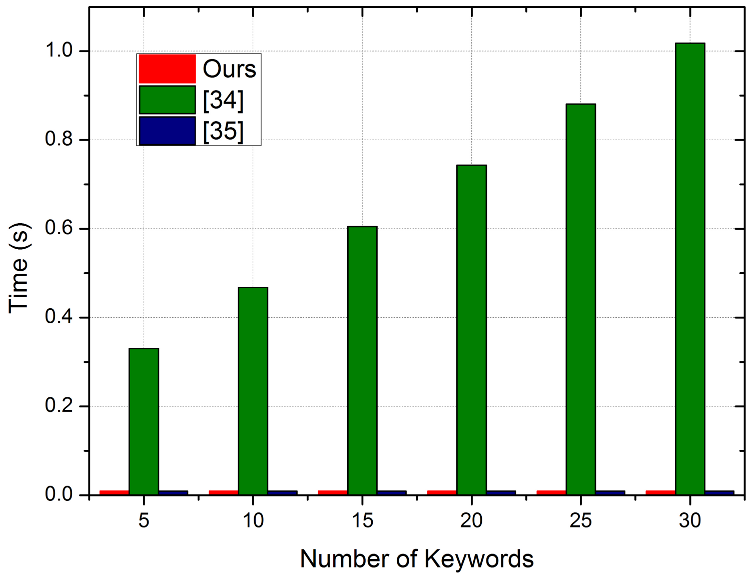

Figure 16: Computation Cost of Keyword Index Generation

Table 15: Computation Cost of Keyword Match Test

$\mathbf{l_1}$	**Ours**	[34]	[35]
5	**0.0031**	0.180	0.657
10	**0.0029**	0.361	0.927
15	**0.0033**	0.541	1.198
20	**0.003**	0.721	1.468
25	**0.0031**	0.901	1.738
30	**0.0034**	1.082	2.009

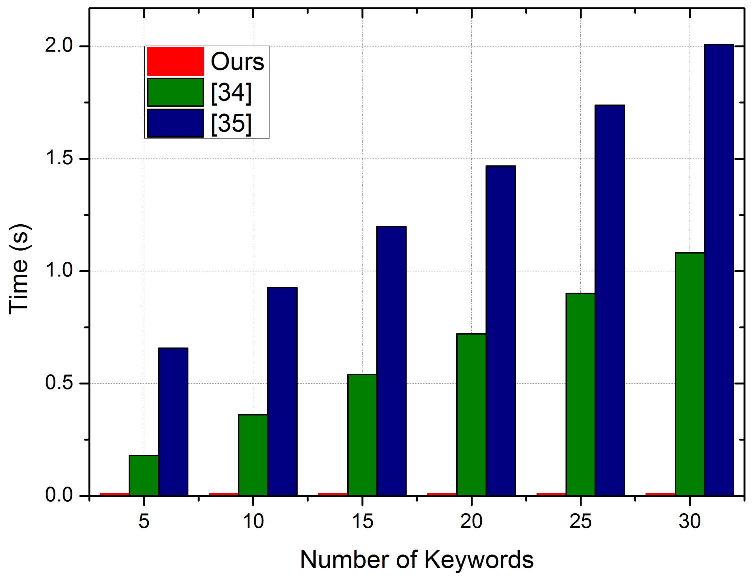

Figure 17: Computation Cost of Keyword Match Test

Fig. 18 shows the computation cost comparison of the non-batch verification algorithm (Algorithm 8) and the batch verification algorithm (Algorithm 9). The value of x-axis in Fig. 18 varies with τ (the number of IoT ciphertexts). When $\tau = 20$, the computation cost in non-batch and batch verification algorithms are 1.089 s and 0.588 s, respectively. When $\tau = 100$, the computation cost in non-batch and batch verification algorithms are 5.445 s and 2.795 s, respectively. Compared with the non-batch verification algorithm, it can be seen that the batch verification algorithm is effective to reduce the computation cost of the patient device.

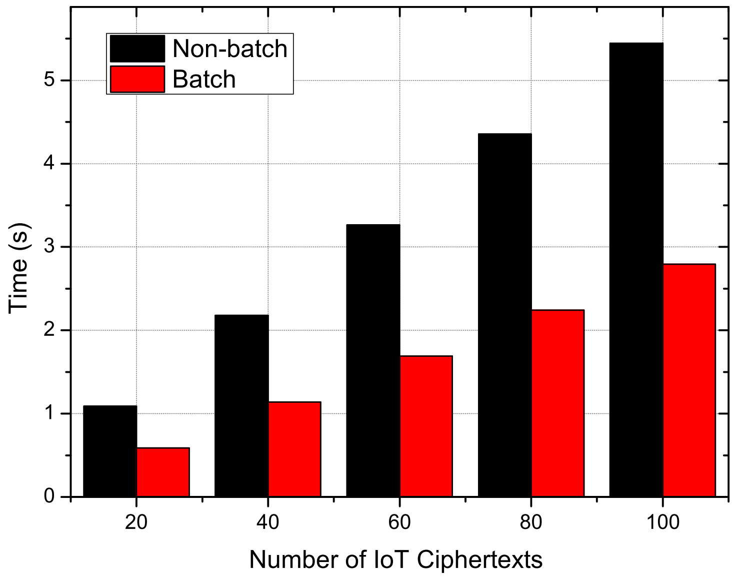

Figure 18: Computation Cost of Batch Verification

9. Conclusion

In this work, we have investigated the privacy-preserving fusion of IoT and big data in the e-health application scenario, and constructed a system to realize secure IoT communication and confidential medical big data storage. The system architecture and security model are defined for the proposed system. A non-interactive and authenticated key distribution procedure is designed for the medical IoT network. A batch authenticated verification algorithm is proposed to verify the source of encrypted IoT messages. Patients' EHRs are encrypted using the ABE methodology to realize access control. We also construct a novel keyword match based access policy updating mechanism to realize fine-grained policy updating control. We compare this system with other schemes and experiments are conducted to evaluate their performances. The testing results demonstrate that this system has performance outperforming the others and applicable in the e-health environment.

Acknowledgement

This work is supported by National Natural Science Foundation of China (No. 61402112, 61672159, 61702105); Open Fund Project of Fujian Provincial Key Laboratory of Information Processing and Intelligent Control (Minjiang University) (No. MJUKF201734); Fujian Major Project of Regional Industry (No. 2014H4015); Major Science and Technology Project of Fujian Province (No. 2015H6013); Technology Innovation Platform Project of Fujian Province (No. 2014H2005), Fujian Collaborative Innovation Center for Big Data Application in Governments, Fujian Engineering Research Center of Big Data Analysis and Processing.

References

[1] Liao D, Sun G, Li H, Yu H, Chang V. The framework and algorithm for preserving user trajectory while using location-based services in IoT-cloud systems[J]. Cluster Computing, 2017, 20(3): 2283-2297.

[2] Sun G, Liao D, Li H, Yu H. Chang V. L2P2: A location-label based approach for privacy preserving in LBS[J]. Future Generation Computer Systems, 2017, 74: 375-384.

[3] Yang Y, Zheng X, Liu X, et al. Cross-domain dynamic anonymous authenticated group key management with symptom-matching for e-health social system[J]. Future Generation Computer Systems, 2017.

[4] Chang V. The big data analysis for measuring popularity in the mobile cloud[J]. 2014.

[5] Sun G, Chang V, Ramachandran M, Sun Z, Li G, Yu H, Liao D. Efficient location privacy algorithm for Internet of Things (IoT) services and applications[J]. Journal of Network and Computer Applications, 2017, 89: 3-13.

[6] Sun G, Xie Y, Liao D, Yu H, Chang V. User-defined privacy location-sharing system in mobile online social networks[J]. Journal of Network and Computer Applications, 2017, 86: 34-45.

[7] Arias O, Wurm J, Hoang K, et al. Privacy and security in internet of things and wearable devices[J]. IEEE Transactions on Multi-Scale Computing Systems, 2015, 1(2): 99-109.

[8] Daz-Snchez D, Sherratt R S, Almenarez F, et al. Secure store and forward proxy for dynamic IoT applications over M2M networks[J]. IEEE Transactions on Consumer Electronics, 2016, 62(4): 389-397.

[9] Li N, Liu D, Nepal S. Lightweight Mutual Authentication for IoT and Its Applications[J]. IEEE Transactions on Sustainable Computing, 2017.

[10] Zhang L, Zhang Y, Tang S, et al. Privacy Protection for E-health Systems by Means of Dynamic Authentication and Three-factor Key Agreement[J]. IEEE Transactions on Industrial Electronics, 2017.

[11] Wu F, Xu L, Kumari S, et al. An efficient authentication and key agreement scheme for multi-gateway wireless sensor networks in IoT deployment[J]. Journal of Network and Computer Applications, 2017, 89: 72-85.

[12] Liu C, Chen J, Yang L T, et al. Authorized public auditing of dynamic big data storage on cloud with efficient verifiable fine-grained updates[J]. IEEE Transactions on Parallel and Distributed Systems, 2014, 25(9): 2234-2244.

[13] Baek J, Vu Q H, Liu J K, et al. A secure cloud computing based framework for big data information management of smart grid[J]. IEEE transactions on cloud computing, 2015, 3(2): 233-244.

[14] Liu C, Ranjan R, Yang C, et al. MuR-DPA: Top-down levelled multi-replica merkle hash tree based secure public auditing for dynamic big data storage on cloud[J]. IEEE Transactions on Computers, 2015, 64(9): 2609-2622.

[15] Yan Z, Ding W, Yu X, et al. Deduplication on encrypted big data in cloud[J]. IEEE transactions on big data, 2016, 2(2): 138-150.

[16] Boneh D, Di Crescenzo G, Ostrovsky R, et al. Public key encryption with keyword search[C]//Eurocrypt. 2004, 3027: 506-522.

[17] Xu P, Jin H, Wu Q, et al. Public-key encryption with fuzzy keyword search: A provably secure scheme under keyword guessing attack[J]. IEEE Transactions on computers, 2013, 62(11): 2266-2277.

[18] Yang Y, Zheng X, Chang V, et al. Semantic keyword searchable proxy re-encryption for postquantum secure cloud storage[J]. Concurrency and Computation: Practice and Experience, 2017, 29(19).

[19] Wang C, Cao N, Ren K, et al. Enabling secure and efficient ranked keyword search over outsourced cloud data[J]. IEEE Transactions on parallel and distributed systems, 2012, 23(8): 1467-1479.

[20] Cao N, Wang C, Li M, et al. Privacy-preserving multi-keyword ranked search over encrypted cloud data[J]. IEEE Transactions on parallel and distributed systems, 2014, 25(1): 222-233.

[21] Cash D, Jaeger J, Jarecki S, et al. Dynamic Searchable Encryption in Very-Large Databases: Data Structures and Implementation[C]//NDSS. 2014, 14: 23-26.

[22] Li H, Liu D, Dai Y, et al. Engineering searchable encryption of mobile cloud networks: When QoE meets QoP[J]. IEEE Wireless Communications, 2015, 22(4): 74-80.

[23] Yang Y, Ma M. Conjunctive keyword search with designated tester and timing enabled proxy re-encryption function for e-health clouds[J]. IEEE Transactions on Information Forensics and Security, 2016, 11(4): 746-759.

[24] Yang Y, Liu X, Deng R H, et al. Flexible Wildcard Searchable Encryption System[J]. IEEE Transactions on Services Computing, 2017.

[25] Goyal V, Pandey O, Sahai A, et al. Attribute-based encryption for fine-grained access control of encrypted data[C]//Proceedings of the 13th ACM conference on Computer and communications security. ACM, 2006: 89-98.

[26] Bethencourt J, Sahai A, Waters B. Ciphertext-policy attribute-based encryption[C]//Security and Privacy, 2007. SP'07. IEEE Symposium on. IEEE, 2007: 321-334.

[27] Waters B. Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization[C]//Public Key Cryptography. 2011, 6571: 53-70.

[28] Ostrovsky R, Sahai A, Waters B. Attribute-based encryption with non-monotonic access structures[C]//Proceedings of the 14th ACM conference on Computer and communications security. ACM, 2007: 195-203.

[29] Yang Y, Liu X, Deng R H, et al. Lightweight Sharable and Traceable Secure Mobile Health System[J]. IEEE Transactions on Dependable and Secure Computing, 2017.

[30] Yang Y, Liu X, Deng R H. Lightweight Break-glass Access Control System for Healthcare Internet-of-Things[J]. IEEE Transactions on Industrial Informatics, 2017.

[31] Yang K, Jia X, Ren K. Secure and verifiable policy update outsourcing for big data access control in the cloud[J]. IEEE Transactions on parallel and distributed systems, 2015, 26(12): 3461-3470.

[32] Li H, Liu D, Alharbi K, et al. Enabling fine-grained access control with efficient attribute revocation and policy updating in smart grid[J]. KSII Transactions on Internet and Information Systems (TIIS), 2015, 9(4): 1404-1423.

[33] Ying Z, Li H, Ma J, et al. Adaptively secure ciphertext-policy attribute-based encryption with dynamic policy updating[J]. Science China Information Sciences, 2016, 59(4): 042701.

[34] Li J, Lin X, Zhang Y, et al. KSF-OABE: outsourced attribute-based encryption with keyword search function for cloud storage[J]. IEEE Transactions on Services Computing, 2017, 10(5): 715-725.

[35] Miao Y, Ma J, Liu X, et al. Attribute-Based Keyword Search over Hierarchical Data in Cloud Computing[J]. IEEE Transactions on Services Computing, 2017.

[36] Yeh L Y, Chiang P Y, Tsai Y L, et al. Cloud-based fine-grained health information access control framework for lightweight iot devices with dynamic auditing and attribute revocation[J]. IEEE Transactions on Cloud Computing, 2015.

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

[38] Lynn B. The Stanford Pairing Based Crypto Library. [Online]. Available: http://crypto.stanford.edu/pbc, accessed May 7, 2014.