Future Generation Computer Systems 115 (2021) 486–496

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

An efficient and outsourcing-supported attribute-based access control scheme for edge-enabled smart healthcare

Hong Zhong, Yiyuan Zhou, Qingyang Zhang, Yan Xu, Jie Cui *

Key Laboratory of Intelligent Computing and Signal Processing of Ministry of Education, School of Computer Science and Technology, Anhui University, China
Anhui Engineering Laboratory of IoT Security Technologies, Anhui University, China
Institute of Physical Science and Information Technology, Anhui University, China

ARTICLE INFO

Article history:
Received 9 January 2020
Received in revised form 24 August 2020
Accepted 19 September 2020
Available online 7 October 2020

Keywords:
Smart healthcare
CP-ABE
Security
Outsourcing capability
Attribute update

ABSTRACT

In recent years, the continuous development of smart healthcare has brought substantial convenience to our lives, especially for emerging edge-enabled smart healthcare systems that transmit and store a large amount of medical data. However, if private health data of users are leaked or tampered with, it will cause enormous damage to users' rights, even threatening the safety of their lives. Traditionally, attribute-based encryption (ABE) is used to encrypt data and implement fine-grained access control for such sensitive information. However, the computing overhead of traditional ABE is relatively large, and in an edge-enabled environment, the outsourcing part of encryption and decryption to the edge node can reduce the computing cost of resource-constrained devices for edge-enabled smart healthcare. However, current schemes are not efficient for resource-constrained devices and edge nodes. Therefore, an efficient ABE scheme is proposed that outsources part of the encryption and decryption to the edge nodes as well as supports attribute updates, enabling flexible right control. A formal security proof is provided, verifying that our scheme is secure under the Decisional Bilinear Diffie–Hellman assumption. The performance of our scheme is evaluated at different security levels, and the experimental results demonstrate that our scheme is more efficient for resource-constrained devices than the traditional ABE.

1. Introduction

The rapid development of the Internet of Things (IoT) [1] has brought many IoT devices into our lives, such as smart healthcare [2–4] and smart home [5], and has significantly improved user convenience. For example, smart healthcare significantly facilitates the interaction between patients and medical staff, *i.e.,* medical staff can obtain the medical records and the latest diagnosis and treatment reports on each patient at any time, as well as quickly diagnose an illness and develop treatment plans at any time. Smart healthcare devices usually use cloud services to store and handle data [6–8]. The medical data being uploaded to an intelligent medical system includes users' private data. If the data is leaked, a significant amount of damage may be caused to users [9]. In general, when data is transmitted through a network, the data needs to be processed to some extent using techniques such as desensitization and encrypted transmission, to protect the security and privacy of the data [10]. However, a large number of computing operations cannot be performed on sensors with limited resources. Recently, a novel computing paradigm (edge computing) was proposed, which is a potential solution to this problem. Edge computing proposes to deploy edge devices around the sensor and process the data, to reduce the network transmission time and to improve the efficiency and latency of data processing [11–13]. Fig. 1 shows the current edge-enabled three-layer smart healthcare architecture, including cloud servers, edge nodes, and sensor devices. For example, in smart healthcare, the ECG data of patients are uploaded to edge nodes on the cloud for processing, and finally the processed data is sent to medical personnel. However, denying unauthorized users the access to private data is a problem to be solved.

In recent years, access control technologies have been proposed as a solution to the problem of unauthorized users illegally accessing protected network resources. However, traditional access control, *i.e.,* role-based access control, cannot guarantee the confidentiality of data storage and a fine-grained access control. In recent years, attribute-based encryption (ABE) has been utilized to improve and implement fine-grained access control to confidential data [14–16]. ABE is a public key encryption technology that binds a series of attributes with a user's identity. By

* Correspondence to: No. 111 Jiu Long Road Hefei, Anhui Province, 230601, China
E-mail address: cuijie@mail.ustc.edu.cn (J. Cui).

https://doi.org/10.1016/j.future.2020.09.021
0167-739X/

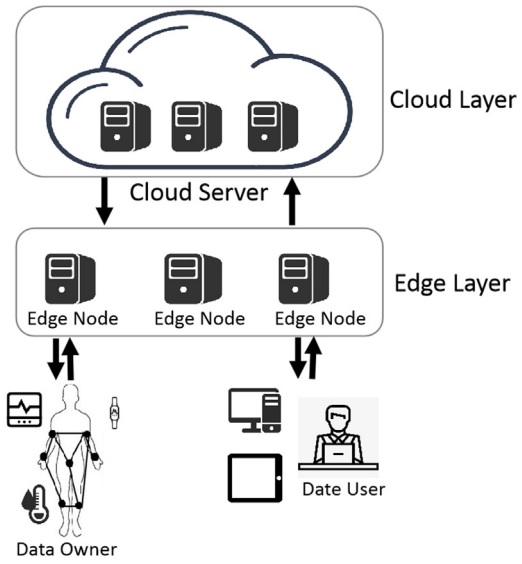


Fig. 1. An example of three-layer smart healthcare architecture.

setting a set of attributes and access structures, only the users having attributes that match to the access structure can gain access through the security of the access structure, *i.e.,* they can obtain the decryption key for the data. This process can realize one-to-many communication and fine-grained access control, which is more suitable for the sharing of healthcare records between doctors and users. However, the primary concern in most ABE schemes is that the operations used (*i.e.,* pairing operation and exponential) in the encryption and decryption algorithms are computationally intensive, which is a significant burden on resource-constrained devices, especially for edge devices and health sensors [17,18]. Performing complex operations on resource constrained sensors and end-users is a challenging problem. In recent years, outsource attribute-based encryption (OABE) has been proposed to mitigate this problem [19,20]. In an edge-enabled environment, partial encryption and partial decryption can be outsourced to the cloud, to reduce the computing load of devices with limited resource.

The OABE scheme entrusts heavy operations to other entities, such as the edge nodes. Users and sensors only need to perform lightweight operations. Compared with the cloud, the edge node is the closest to the information sensor and the user, which can partially encrypt and partially decrypt data more effectively and quickly. In addition, the calculation load of the sensor is reduced. However, most of the current outsourcing encryption and decryption schemes are not sufficiently effective and cannot complete the attribute update operation. Even if some load are outsourced to the edge nodes, the remaining load on data owners and data users linearly increases with the number of attributes and is still a heavy burden for resource constrained devices [17].

Therefore, we need to consider maintaining the load that the data owner and data user need to calculate within an acceptable range.

In this paper, we propose an efficient and attribute-based access control scheme supported by outsourcing, for edge enabled smart healthcare. Compared with the traditional attribute encryption scheme, we outsource partial encryption and decryption operations to the edge nodes, to reduce the computing load of resource-constrained devices and to protect the privacy and security of medical data. At the same time, the scheme supports attribute updating, which improves the efficiency of the scheme. Our scheme makes the following contributions:

- We studied a scenario involving novel edge-enabled smart healthcare and adapted the traditional ABE with outsourcing encryption and decryption support.
- We improved the efficiency of the OABE scheme, reducing the burden of resource-constrained devices, such as sensors. This makes our scheme more suitable for smart healthcare. In addition, our scheme supports attribute updates, thus increasing the security of medical data.
- We proved the security of the proposed scheme under the Decisional Bilinear Diffie–Hellman (DBDH) assumption, and the experimental results show that the proposed scheme is efficient for smart healthcare.

The remainder of this paper is organized as follows. In Section 2, we discuss related works. Section 3 describes the use of preparatory knowledge. We introduce the security model in Section 4, followed by the proposed scheme in Section 5. Then, we prove the security of our scheme based on the security model and the DBDH assumption in Section 6. In Section 7, we analyze the performance of our scheme. Finally, we summarize of our study and discuss future work in Section 8.

2. Related work

In this section, we reviewed a series of related works, in terms of cloud-based and edge-based smart healthcare systems, ABE-based access control schemes, and outsourcing ABE-based access control schemes.

2.1. Smart healthcare systems

In recent years, the development of smart healthcare has also been quite rapid. Based on the cloud model, many smart healthcare systems have been proposed gradually [21,22]. The system improves the performance of a smart healthcare system. For example, Zhang et al. [23] proposed a four-layer and patient-centered system for the cloud-based smart healthcare system, including data collection, data management, parallel computing, and data-oriented service. In recent years, edge computing has become increasingly popular [11,24]. The edge node is the closest to the limited resource devices and has strong computing power, and some edge-based smart healthcare systems are proposed. Zhang et al.had summarized some video-based smarted healthcare systems in [25], where video analytics is a typical application domain. Wu et al. [26] created an emergency medical service to provide rapid medical treatment for patients in need of first aid. Their medical data will be uploaded to the public platform in advance so that the hospital can prepare to improve the efficiency of first aid before patients' arrival. However, the above systems cannot meet the security requirements well. To realize the calculation of security encryption in intelligent medical treatment, Sun et al. [27] encrypted the health data onto full homomorphic encryption (FHE). To solve the problems of privacy and efficiency, Cai et al. [28] introduced a novel medical record of the mobile cloud to enhance information security without compromising too much performance. However, these systems didn't provide access control for these medical records.

2.2. Attribute-based encryption

With the development of smart healthcare [29], people increasingly value their personal information not only the security of their biological characteristics [30,31] but also the security of personal information data. To better protect data privacy and realize communication between medical staff and patients, only relevant medical staff can access private data. Fine-grain access control is required to achieve this. Thus, some access control

schemes had been proposed [19,32] for the smart healthcare system. For example, Chen et al. [32] provided a new role-based access control that could be used for medical resource information data. Medical data could be tracked and further authorized access could be made to the system resources. Only authorized users could access it. With the development of cloud computing and big data, the ABE scheme is gradually emerging and has more flexible data access control. Typically, the ABE scheme is divided into Key-Policy Attribute-based Encryption (KP-ABE) [15] and Ciphertext-Policy Attribute-based Encryption (CP-ABE) [33]. In [34], Narayan et al.proposed a privacy-protection access control scheme based on ABE for patients. Attrapadung et al.proposed a constant ciphertext size KP-ABE scheme [35] with a non-monotonic access structure. Water et al. [36] proposed a CP-ABE scheme which can allow the attribute formula for the encryption program to specify the access control to realize an efficient access control strategy.

2.3. Outsourcing attribute-based encryption

As the emerging of edge computing, Dash et al. [37] proposed a method on how to make the security algorithm more effective in smart healthcare by fog computing and edge computing. Thus, we can improve the efficiency of the system by outsourcing encryption and decryption to the edge nodes. Using the ABE-based scheme for cloud storage service can not only ensure the security of data but also provide fine-grained access control of data. However, the computation cost of ABE is usually very large, including many pairing operations and exponential operations. Therefore, this greatly limited their use of resource-constrained devices (such as information sensors or mobile devices). The ABE computation complexity will increase linearly as the number of attributes associated with ABE [38]. In order to reduce the computing burden of resource-constrained devices, some schemes suggested that some decryption operations could be outsourced to cloud servers or other proxy nodes. At present, most of the outsourcing schemes are about outsourcing the decryption parts with the cloud server. The outsourcing decryption scheme is proposed for some CP-ABE schemes [19]. Yao et al. [39] proposed a novel outsourcing ABE system that can support outsourcing key generation and outsourcing decryption and can verify the results returned by the third party. Liu et al. [40] proposed an OABE scheme for outsourcing decryption, besides, it also supports attribute revocation and policy updating. In the smart healthcare system, the OABE scheme of outsourcing encryption and decryption to edge nodes is more efficient than the OABE scheme on the cloud.

3. Preliminary

3.1. Bilinear pairings

Let G_1 and G_2 be two groups with the same prime order p, while g_1 and g_2 are the generator for the groups of G_1 and G_2, respectively. Z_p refers to the prime p-order cyclic group. In addition, a bilinear mapping e: $G_1 \times G_1 \rightarrow G_2$ is chosen with the following properties:

(1) Bilinearity: $\forall g_1, g_2 \in G_1$ and $a, b \in Z_p$, among them, a and b are elements in Z_p, $e(g_1^a, g_2^b) = e(g_1, g_2)^{ab}$.

(2) Non-degeneracy: Exists $g_1, g_2 \in G_1$, that $e(g_1, g_2) \neq 1$.

(3) Computability: For each $g_1 \in G_1$, $g_2 \in G_1$, $e(g_1, g_2)$ is an admissible algorithm.

3.2. Decisional bilinear Diffie–Hellman assumption

Let g be the generators of a group G. Given random values $x, y, z \in Z_p$ and a bilinear mapping e: $G_1 \times G_1 \rightarrow G_2$, while G_2 is a group, as well as have element $L \in G_2$. The DBDH assumption is that no probabilistic polynomial-time algorithm Q can distinguish with more than a negligible advantage between tuple $(g, g^x, g^y, g^z, e(g, g)^{xyz})$ and tuple (g, g^x, g^y, g^z, L), while the advantage δ of algorithm Q is as follows:

$$\delta = |\Pr[Q(g, g^x, g^y, g^z, e(g, g)) = 0] - \Pr[Q(g, g^x, g^y, g^z, L) = 0]| \quad (1)$$

3.3. Access tree

In our paper, the access structure is described as an access tree T. Each leaf node of T represents one attribute, and each non-leaf node represents a threshold gate. num_a represents the number of child nodes of a node, and$k_a$represents the number of the threshold value, while $0 < k_a \leq num_a$. In this case, the non-leaf could hide the security by some schemes, i.e., security sharing with (k_a, num_a) threshold, and can be recovered by k_a child nodes. Among them, the k_a value of leaf nodes is 1. In addition, some functions are defined to describe the access tree. $Parent(n)$ being used to get the parent of the node n. $index(n)$ represents the order value of a node n when calculating the value of $Parent(n)$. $attr(n)$ represents the attribute of the leaf node n. Typically, the structure of a parent node n with some child nodes is structured by a random polynomial while the polynomial order is the threshold k_n, and $d_n = k_n$- 1. When the attributes owned by the user satisfy the attributes of access tree T, the secret value of the node can be obtained. When the user's attribute does not satisfy the access tree T, the user cannot decrypt the ciphertext. For example, assume the threshold value of the access tree is 2, and the leaf node attribute of the access subtree is $\{A_1, A_2, A_3\}$. When the user's attribute is $\{A_1, A_4, A_5\}$, it does not satisfy any two attributes of the access tree, the secret value cannot be recovered by linear secret sharing. If the attributes of the user satisfy the attributes of the access tree, the secret value of the root node can be finally accessed. Thus, the final plaintext can be obtained by decrypting the ciphertext with the private key.

4. System model and security model

4.1. System model

Fig. 2 introduces the main structure of our scheme, including the Cloud Server, Edge Node, Key Authority (KA), Data Owner (DO), and Data User (DU). The cloud layer contains remote cloud servers and the edge nodes and KA belongs to the edge layer, while the DO and DU belong to the IoT layer. The roles played by the five participants in the scheme are as follows.

Cloud Service: The cloud server is a trusted party, providing storage functions. It is used to receive the ciphertext from the edge node, and then store and send the ciphertext to the edge node. Furthermore, the cloud also performs ciphertext updating when related attributes are updated.

Edge Node: The edge node receives the locally encrypted ciphertext from the data owner, performs further encryption operations, then, it sends the encrypted ciphertext to the cloud service. Other edge nodes also receive the ciphertext from the cloud server and performing partial decryption operations, but it cannot get any content in the ciphertext. Edge nodes can be gateways, mobile phones, etc.

Key Authority (KA): In our system, KA is a fully trusted entity. KA is responsible for managing the setting of global parameters and the ability to send public keys and private keys. KA is also

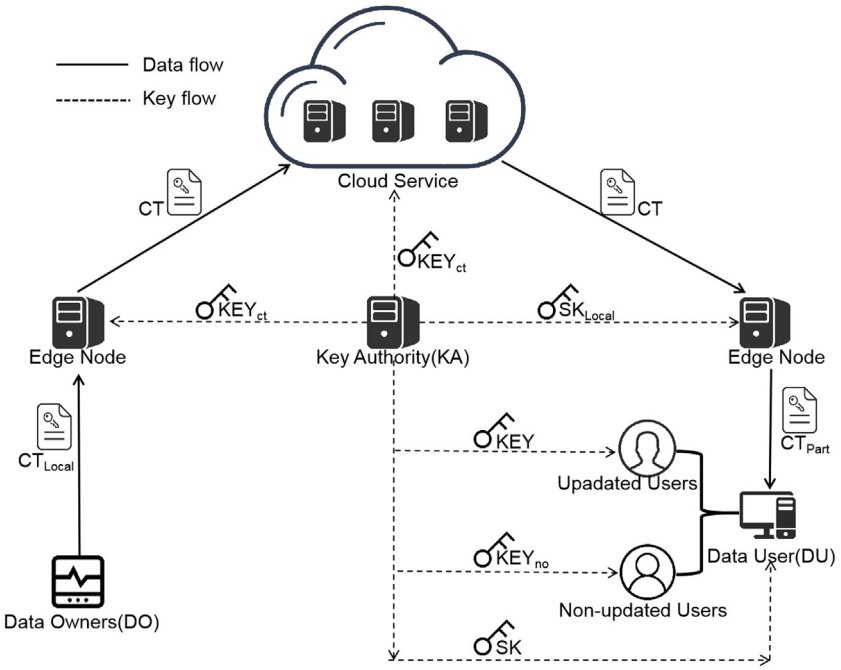


Fig. 2. System architecture.

responsible for the production of system parameters and the registration of users, by generating private key *SK* for each user. KA generates the update key KEY_{ct} for the cloud service, the update key $KEY_{j\to o}$ for the users that have already updated, and the update key KEY_{no} for the users that have not.

Data Owner (DO): DO first needs to implement a local encryption algorithm for the message, after which the encrypted message will be uploaded to the nearby edge node. DO is usually an information sensor, such as an ECG machine, etc.

Data User (DU): Because DU is usually a device with resource-constrained. After obtaining the partially decrypted ciphertext from edge nodes, DU performs remained decryption operations. DU is usually terminal equipment, such as a computer, mobile phone, etc.

In our system, TA will first run the *Setup* algorithm, generate the master key *MK* and the public key *PK*, and send the *PK* to DO. Then KA runs the *KeyGeneration* algorithm to generate the local private key SK_{local} and private key *SK*. *SK* is sent to the edge node and SK_{local} is sent to DU. Next, DO runs the *LocalEncryption* encryption algorithm to generate part of the ciphertext CT_{local}, and sends the CT_{local} to the edge node. The edge node executes the *OutsourcedEncryption* algorithm, generates the ciphertext *CT* and sends the ciphertext *CT* to the cloud service. The cloud service sends the *CT* to another edge node. The edge nodes run the $Decrypt_{out}$ algorithm to decrypt the *CT*, output the CT_{part}, and send it to DU. DU runs the $Decrypt_{local}$ algorithm to get message *M*.

4.2. Definition

$Setup(\lambda)\to PK, MK$: First, KA runs the *setup* algorithm. A security parameter λ is inputted and the *setup* algorithm output the public key *PK* and the master key *MK*.

$KeyGeneration(S, MK)\to SK_{local}, SK$. This algorithm is used to generate a private key for the user. The algorithm inputs the attribute set *S* and the master key *MK*. Then, the outsourcing key *SK* and the local private key SK_{local} are output.

$LocalEncryption(M, PK, T)\to CT_{local}$: In order not to let the edge node know the content of the message, first, DO run a local encryption algorithm to encrypt message *M*. This local encryption algorithm inputs the access structures *T*, as well as the master key *PK* and the Message *M*, Then the algorithm outputs part of the ciphertext CT_{local}.

$OutsourcedEncryption(T, C_{local}, PK)\to CT$: After receiving some encrypted ciphertext, the edge node encrypts it in the next step. In the outsourcing encryption algorithm, we need to input access structure *T*, public key *PK*, partial ciphertext C_{local}. Finally, the outsourcing encryption algorithm outputs ciphertext *CT*.

$Decrypt_{out}(CT, SK)\to CT_{part}$: After the edge node decrypts the ciphertext partially to reduce the computing load of resource-constrained devices. The algorithm inputs ciphertext *CT* and private key *SK*, and outputs CT_{part}.

$Decrypt_{local}(CT_{part}, SK_{local})\to M$: After the DU receives the partially decrypted ciphertext, if the owned attribute-related private key matches the access tree *T*, then the message *M* can be obtained by running the $Decrypt_{local}$ algorithm. First, the algorithm inputs part of ciphertext CT_{part} and local private key SK_{local}, then outputs the output message *M*.

$UpdateKeyGen(PK, MK, SK, m_j, m_o)\to(KEY_{j\to o}, KEY_{no}, KEY_{ct})$: In order to update the user's attributes, the old attribute m_j is assuming to be updated with a new attribute m_o. We call m_j the attribute to be updated, and the user who updates the containing attribute from m_j to m_o is called the updated user. Users with m_j attribute that are not-updated are called not-updated users. The update key can be obtained by running the UpdateKeyGen algorithm through KA. By inputting public key *PK*, master key *MK*, outsourcing private key *SK*, and attributes m_j and m_o, the algorithm outputs the update private key of the user's private key $KEY_{j\to o}$, KEY_{no}, KEY_{ct}.

$UpdateSK_1(SK, KEY_{j\to o})\to SK_{update}$: This update algorithm is run by users who have already been updated. The algorithm updates and outputs the private key SK_{update}, with the inputs of the key *SK* and the update key $KEY_{j\to o}$.

$UpdateSK_2(SK, KEY_{no})\to SK_{unupdate}$: This update algorithm is run by users who are not updated. The algorithm inputs the private key *SK*, update the key KEY_{no}, and outputs the updated private key $SK_{unupdate}$.

$UpdateCT(CT, KEY_{ct})\to CT_{update}$: This ciphertext update algorithm is run by the Cloud Server. Ciphertext *CT* and updates key KEY_{ct} are inputting first, followed by the output update ciphertext CT_{update}.

4.3. Security model

In this model, an adversary can query the secret key of an attribute set S and get any public key, but these private keys cannot be directly used to decrypt the challenge's ciphertext. We provide the following formal definition as:

Init. First, we define the adversary as $\mathcal{A}$ and the challenger as $\mathcal{C}$. Adversary $\mathcal{A}$ sends an access structure $\mathcal{T}$ to Challenger $\mathcal{C}$.

Setup. The *Setup* algorithm is executed by Challenger $\mathcal{C}$ and then the public key PK is sent to Adversary $\mathcal{A}$.

Phase 1. Adversary $\mathcal{A}$ can adaptively send any attribute set to Challenger $\mathcal{C}$, and adversary $\mathcal{A}$ can query the private key of attribute set $\mathcal{S}$ (The attribute set queried by $\mathcal{A}$ cannot match the access structure $\mathcal{T}$). Then, for attribute set $\mathcal{S}$, Challenger $\mathcal{C}$ runs the *KeyGeneration* algorithm and sends the private key SK corresponding to the attribute set to adversary $\mathcal{A}$. When his attribute set needs to be updated, adversary $\mathcal{A}$ will also query the update key (The new attribute set does not satisfy the requirement of access structure T). In this case, Challenger $\mathcal{C}$ runs the *UpdateKeyGen* algorithm and sends the updated key associated with the new attribute set to adversary $\mathcal{A}$.

Challenge. Adversary $\mathcal{A}$ submits two messages of equal length M_0, M_1 to challenger $\mathcal{C}$. Challenger $\mathcal{C}$ flips a random coin b and encrypts M_b with access structure $\mathcal{T}$ by running algorithm Local encryption and outsourced encryption to generate the ciphertext CT. Ciphertext CT is sent to adversary $\mathcal{A}$.

Phase 2. Adversary $\mathcal{A}$ can query more update keys and private keys for other attribute sets (But in this case, the update keys and private keys cannot decrypt ciphertext CT).

Guess. Adversary $\mathcal{A}$ output a guess b' of b. In this game, the advantage of adversary $\mathcal{A}$ is defined as:

$$Adv(\mathcal{A}) = |\Pr(b' = b) - \frac{1}{2}| \tag{2}$$

5. Our construction

In our construction, we elaborate on the setup, key Generation, local encryption, outsourcing encryption, outsourcing decryption, local decryption, attribute update algorithms defined above.

5.1. Setup

$Setup(\lambda) \rightarrow PK, MK$: First, the implicit parameter λ needs to be input. Define the universe of attributes $P = \{m_1, m_2, \ldots, m_n\}$. Let G_1 be a bilinear group of the prime order q, and let g be a generator of G_1. Let bilinear map e: $G_1 \times G_1 \rightarrow G_2$. Define a hash function F: $\{0,1\}^* \rightarrow G_1$ and the Lagrange coefficient $\Delta_{i,P}(x) = \prod_{p\in P, p\neq i} \frac{x-p}{i-p}$ for any $i \in Z_p$. Then algorithm randomly selects two integers α, $\beta \in Z_p$, and for each $m_j \in P$, selects a random $n_j \in Z_p$, then let $PK_j = g^{n_j}$. Finally, the algorithm outputs $PK = \left(G1, H(\cdot), g, h = g^{\beta}, e(g,g)^{\alpha}, \left(PK_j = g^{n_j} | m_j \in P\right)\right)$, and the master key $MK = (\beta, g^{\alpha}, (n_j | m_j \in P))$.

5.2. Key generation

$KeyGeneration(S, MK) \rightarrow SK_{local}, SK$: This algorithm selects two random r, $t \in_R Z_q$, then for each attribute $j \in S$, selects $r_j \in_R Z_q$. The algorithm outputs the user local private key $SK_{local} = (d = g^{\frac{\alpha+r}{\beta}}, t)$ and private key $SK = (d = g^{\frac{\alpha+r}{\beta}}, \forall m_j \in S : D_{j0} = (g^{rn_j^{-1}} H(j)^{r_j})^t, D_{j1} = g^{r_j t n_j^{-1}})$.

5.3. Outsourcing encryption

This phase consists of two algorithms: Local encryption and Outsourcing encryption.

$LocalEncryption(M, PK, T) \rightarrow CT_{local}$: This algorithm is used to encrypt the message M with the access tree T, while $M \in G_2$. For the root node R, the algorithm selects a random value $s \in_R Z_q$. T_1 and T_2 are subtrees of T and $T = T_1 \cup T_2$. The algorithm chooses a 1-degree polynomial q(·) that $q_R(0) = s$, and selects $s_1, s_2 \in_R Z_q$, let $q_R(1) = s_1$, $q_R(2) = s_2$. The T_2 includes only one virtual attribute. Then, the algorithm outputs partial ciphertext $C_{local} = (C^{\sim} = Me(g,g)^{\alpha s}, C = h^s, \forall y \in Y_2: C_y = g^{q_y(0) n_j^{-1}}, C'_y = H(att(y))^{q_y(0) n_j})$.

$OutsourcedEncryption(\mathrm{T}, C_{local}, PK) \rightarrow CT$: Firstly, x is a node of the sub-tree T_1. For each node $x \in T_1$, the algorithm chooses a polynomial q_x, starting from the root node R_1 of the sub-tree T_1 and selects polynomial in a top-down fashion to build that access tree T_1. The selected polynomial q_x is with the degree d_x according to the definition in T_1, as mentioned in Section 3, while $d_x = k_x$- 1, performing a (k_x, num_x) threshold. Note that the value of R_1 has been calculated before as $q_{R_1}(x) = s_1$. The value of child node c of parent node $parent(c)$ is calculated by the equation as $q_c(0) = q_{parent(c)}(index(x))$, and the algorithm randomly chooses coefficients to build the polynomial q_x. The leaf node is the attribute node and the attribute set is indicated as Y_1. Then, the algorithm outputs $CT_{out} = (T_1, \forall y \in Y_1 : C_y = g^{q_y(0) n_j}, C'_y = H(att(y))^{q_y(0) n_j})$.

The whole ciphertext is given as follows: $CT = (\mathcal{T}, C^{\sim} = Me(g,g)^{\alpha s}, C = h^s, \forall y \in Y : C_y = g^{q_y(0) n_j}, C'_y = H(att(y))^{q_y(0) n_j})$.

5.4. Outsourcing decryption

For decrypted users whose computing power is limited, the algorithm chooses outsourcing decryption to reduce their computing load. This phase includes the following two algorithms: $Decrypt_{out}$ and $Decrypt_{local}$.

$Decrypt_{out}(CT, SK) \rightarrow CT_{part}$: Our decryption algorithm is a recursive algorithm. If the attribute set of the user meet the access structure T, then using the algorithm, the user can successfully decrypts the ciphertext. If the y is a leaf node, let $i = \mathrm{att}(y)$. When y does not exist in the user's private key, the node outputs $DecNode(CT, SK, y) = \perp$. If a user has attribute y in his private key, then the algorithm calculates the algorithm:

$$\begin{aligned} DecNode(CT, SK, y) &= \frac{e(C_y, d_{i0})}{e(d_{i1}, C'_y)} \\ &= \frac{e(g^{q_y(0) n_j}, g^{rtn_j^{-1}} H(j)^{n_j t})}{e(g^{r_j t n_j^{-1}}, H(att(y)))^{q_y(0) n_j}} \\ &= e(g,g)^{rtq_y(0)} \end{aligned} \tag{3}$$

When y is a non-leaf node, for each child node c, it performs the recursive algorithm $DecryptNode(CT, SK, c)$ to obtain the value and save that as F_c in the set S_y. The algorithm could recover the value F_y of y by calculated as follows:

$$\begin{aligned} F_y &= (\prod_{c\in S_y} F_c)^{\Delta i, S_y(0)} \\ &= (e(g,g)^{\sum_{c\in S_y} rq_c(0)t})^{\Delta i, S_y(0)} \\ &= e(g,g)^{\sum_{c\in S_y} trq_{parent(c)}(index(c))\Delta i, S_y(0)} \\ &= e(g,g)^{rt \sum_{c\in S_y} q_y(i)\Delta i, S_y(0)} \\ &= e(g,g)^{rtq_y(0)} \end{aligned} \tag{4}$$

Finally, ciphertext can be obtained: $CT_{part} = (T, C^{\sim} = Me(g,g)^{\alpha s}, C = h_s, F_R)$.

$Decrypt_{local}(CT_{part}, SK_{local}) \rightarrow M$: In this algorithm, the message M is obtained by inputting CT and SK. The algorithm for the fully decrypting ciphertext is as follows:

$$\frac{\widetilde{C}(F_R)^{\frac{1}{t}}}{e(C,d)} = \frac{\frac{Me(g,g)^{\alpha s}}{e(h^s, g^{\frac{\alpha+r}{\beta}})}}{e(g,g)^{rs}} = M \tag{5}$$

5.5. Attribute update

5.5.1. Update key generation

UpdateKeyGen(PK, MK, SK, m_j, m_o) $\rightarrow$ ($KEY_{j\rightarrow o}$, KEY_{no}, KEY_{ct}). This algorithm generates the update key $KEY_{j\rightarrow o}$, KEY_{no} and KEY_{ct} to update attribute m_j with m_o, by entering PK, MK, SK, and attributes m_j, m_o. The detail of this algorithm is described as follows: Firstly, the algorithm generates the private key $KEY_{j\rightarrow o}$ by the equation $KEY_{j\rightarrow o} = n_j/n_o$, which is used to update the user's private key SK. Secondly, the algorithm selects a random number $n_v \in Z_p (n_j \neq n_v)$ for each user who does not have the attribute m_j. Here $KEY_{no} = n_j/n_v$. In addition, the KEY_{ct} is generated to update ciphertext where $KEY_{ct} = n_v/n_j$. Finally, the public key of new attribute m_o is $PK_o = (PK_j)^{KEY_{ct}} = (g^{n_j})^{n_v/n_j} = g^{n_v}$.

5.5.2. Private key update for updated users

$UpdateSK_1(SK, KEY_{j\rightarrow o}) \rightarrow SK_{update}$: When the user receives the update key $KEY_{j\rightarrow o}$ used to update the private key SK, the user's private key can be updated by using the following algorithm 1 as shown below.

Algorithm 1: *$UpdateSK_1$*

Input: SK,$KEY_{j\rightarrow o}$
Output: Some parameters of SK_{update}
forall $\forall m_i \in S\backslash\{m_j\}$ **do**
  $D_{i0} = (g^{rn_i^{-1}} H(i)^{r_i})^t$;
  $D_{i1} = (g)^{r_i t n_i^{-1}}$;
end
$D_{o0} = (g^{rn_j^{-1} KEY_{j\rightarrow o}} H(o)^{r_o})^t$;
$D_{o1} = (g)^{r_o t n_j^{-1} KEY_{j\rightarrow o}}$;
$SK_{update} = \{d, \forall m_i \in S\backslash\{m_j\}: D_{i0}, D_{i1}; D_{o0}, D_{o1}\}$

5.5.3. Private key update for not-updated users

$UpdateSK_2(SK, KEY_{no}) \rightarrow SK_{unupdate}$: When each non-updated user receives an update key KEY_{no} to update their private key SK, the user's private key can be updated by using the following algorithm 2.

Algorithm 2: *$UpdateSK_2$*

Input: SK,KEY_{no}
Output: Some parameters of $SK_{unupdate}$
forall $\forall m_i \in S\backslash\{m_j\}$ **do**
  $D_{i0} = (g^{rn_i^{-1}} H(j)^{r_i})^t$;
  $D_{i1} = (g)^{r_i t n_i^{-1}}$;
end
$D_{j0} = (g^{rn_j^{-1} KEY_{no}} H(j)^{r_j})^t$;
$D_{j1} = (g)^{r_j t n_j^{-1} KEY_{no}}$;
$SK_{unupdate} = \{d, \forall m_i \in S\backslash\{m_j\}: D_{i0}, D_{i1}; D_{j0}, D_{j1}\}$

5.5.4. Update of the ciphertext

$UpdateCT(CT, KEY_{ct}) \rightarrow CT_{update}$: When new users join, we need to ensure that they can decrypt the original ciphertext, so we need to update the previous ciphertext. For attribute n_j to be updated, we use the following algorithm 3 to update it.

Algorithm 3: *$UpdateCT$*

Input: CT, KEY_{ct}
Output: Some parameters of CT_{update}
forall $\forall m_i = att(y) \in Y$ **do**
  if $m_i = m_j$ **then**
    $E_{y0} = g^{q_y(0) n_i}$;
    $E_{y1} = H(att(y))^{q_y(0) n_i}$;
  else
    $E_{y0} = g^{q_y(0) n_i KEY_{ct}}$;
    $E_{y1} = H(att(y))^{q_y(0) n_i KEY_{ct}}$
  end
end
$CT_{update} = \{\mathcal{T}, \widetilde{C}, C, \forall m_i = att(y) \in Y, E_{y0}, E_{y1}\}$

6. Security proof

In this section, we will prove that the proposed scheme is secure under the assumption of DBDH.

Theorem 1. *Suppose there exists a probabilistic-polynomial time adversary that can break our scheme with an ignored advantage that is the advantage $\varphi > 0$. At the same time, there is also a probabilistic-polynomial time algorithm Q, which can distinguish a DBDH tuple from a random tuple with the advantage of $\frac{\varphi}{2}$.*

Proof. Let G_1 be a bilinear group of prime order q, and its generator is g. Let bilinear map e: $G_1 \times G_1 \rightarrow G_2$. First, Challenger $\mathcal{C}$ randomly selects $x, y, z \in Z_p$, a random element $F \in G_2$ and flips a fair binary coin b. If $b = 0$, challenger $\mathcal{C}$ sets $(g, X, Y, Z, L) = (g, g^x, g^y, g^z, e(g,g)^{xyz})$, where $L = e(g,g)^{xyz}$; otherwise it set $(g, X, Y, Z, L) = (g, g^x, g^y, g^z, F)$, where $L = F$. In the next game, we set up algorithm δ to play the role of Challenger $\mathcal{C}$.

Init. First, Adversary $\mathcal{A}$ selects an access structure T^* and declares that it wants to be challenged and then sends the access structure T^* to Q.

Setup. Q need to provides public key PK to adversary $\mathcal{A}$. Q runs the setup algorithm, generates a random $\alpha' \in Z_p$ and calculates $\alpha = \alpha' + xy$. Then it set $u = e(g,g)^{\alpha} = e(g,g)^{\alpha'} e(g,g)^{xy}$, $h = g^{\beta} = g^y = Y$. For each $m_j \in P$, Q choose a random s_j and computes $PK_j = g^{\beta s_j^{-1}} = g^{n_j}$. If $m_j \in T^*$, then $n_j = \beta s_j^{-1}$; otherwise, where $s_j = n_j$, then $PK_j = g^{s_j} = g^{n_j}$. Then, Q sends the public key $PK = \{u, h, \ PK_j \big| m_j \in P\}$ to adversary $\mathcal{A}$.

Phase 1. In the stage, adversary $\mathcal{A}$ submits any attribute set to Q, and then Q runs the *KeyGeneration* algorithm to generate private key SK and send it to $\mathcal{A}$. First, Q randomly selects an $r' \in Z_p$, and then it calculates $r = r' - xy$. Then we can get $D = g^{\frac{\alpha+r}{\beta}} = g^{\frac{\alpha'+xy+r'-xy}{\beta}} = g^{\frac{\alpha'+r'}{y}}$. For each $m_j \in S$, if $m_j \in T^*$, Q compute $D_{j0} = (g^{r'\beta^{-1}s_j} H(j)^{r_j})^t = (g^{r' n_j^{-1}} H(j)^{r_j})^t$, $D_{j1} = g^{r_j t \beta^{-1} s_j} = g^{r_j t n_j^{-1}}$; otherwise, $D_{j0} = (g^{r' s_j^{-1}} H(j)^{r_j})^t = (g^{r' n_j^{-1}} H(j)^{r_j})^t$, $D_{j1} = g^{r_j t s_j^{-1}} = g^{r_j t n_j^{-1}}$. Then it can get the value of the private key SK and value t and send them to adversary $\mathcal{A}$. For the old attribute m_j, a new attribute m_o will be generated to update that. and the condition must be met so that the new attribute o does not meet T^* at the same time. Then, Q sets the update key as $KEY_{j\rightarrow o} = n_j/n_o$.

Challenge. First, adversary $\mathcal{A}$ submits two messages M_0, M_1 and sends them to Q. Q sends T^* to the DO, and then the DO randomly selects $z \in_R Z_q$, and z_1, $z_2 \in_R Z_q$. Then calculate $C^{\sim} = M_b e(g,g)^{\alpha z} = M_b e(g,g)^{(\alpha'+xy)z} = M_b L e(g,g)^{\alpha' z}$, $C = h^z = g^{\beta z} = Z g^{\beta}$. Q sends T^* to edge node to construct v_i of s for all related attribute n_j. At the same time, v_i is randomly selected

Table 1
Functional comparison

Scheme	OABE-E	OABE-D	OABE-ED	Our scheme
Outsourcing encryption	✓	×	✓	✓
Outsourcing decryption	×	✓	✓	✓
Attribute update	×	×	×	✓

from ${}_RZ_q$. Edge node compute $C_i = g^{n_i v_i}$, $C_i' = H(att(i))^{n_i v_i}$. The edge node then returns these values to Q. Finally, Q gets $CT = (T^*, C^{\sim} = M_b e(g,g)^{\alpha' z}, C = h^z = Zg^{\beta}, \{C_i = g^{n_i v_i}, C_i' = H(att(i))^{n_i v_i} | \forall m_j \in T^*\})$. Then Q sends the CT to adversary $\mathcal{A}$.

Phase 2. The same as Phase 1.

Guess. Adversary $\mathcal{A}$ submits a guess b′ of b. if b′ $= b$, then Q putouts 0, which means $T = e(g,g)^{xyz}$; otherwise, $T = F$ and Q outputs 1. If $T = e(g,g)^{xyz}$, then CT is an available ciphertext. At the same time, the advantage of $\mathcal{A}$ is φ. Therefore, $\Pr[Q(g, g^x, g^y, g^z, L = e(g,g)^{xyz}) = 0] = \frac{1}{2} + \varphi$. If $T = f$, Then ciphertext CT is completely randomly selected, then $\Pr[Q(g, g^x, g^y, g^z, L = F) = 0] = \frac{1}{2}$. Thus, in this game, the advantage δ of algorithm Q can be expressed in the following form.

$$\begin{aligned}\delta &= \frac{1}{2}\Pr[Q(g,g^x,g^y,g^z,L=e(g,g)^{xyz})=0] \\ &+ \Pr[Q(g,g^x,g^y,g^z,L=F)=0] - \frac{1}{2} \\ &= \frac{1}{2}(\frac{1}{2}+\varphi+\frac{1}{2}) - \frac{1}{2} \\ &= \frac{\varphi}{2}\end{aligned} \tag{6}$$

7. Performance analysis

In this part, we compare our scheme with the other three schemes, Li et al. [41], Li et al. [42], and Asim et al. [43] respectively. Then, we will compare our scheme with other schemes from the following two aspects. The functions and the computation overhead of the scheme are analyzed respectively. It mainly focuses on the function of the scheme, whether it supports outsourcing encryption, outsourcing decryption, and update function. The encryption time and decryption time of each scheme are compared.

To simplify, we label the Li et al.'s schemes [41] supporting outsourcing encryption as OABE-E, label Li et al.'s schemes [42] as OABE-D, which support outsourcing decryption ABE which checkability, and label Asim et al. [43] as OABE-ED that supports outsourcing encryption and decryption.

7.1. Functions

The comparison from the function view is concluded in Table 1. The OABE-E scheme can only support the function of outsourcing encryption, OABE-D can only support the outsourcing decryption, and OABE-ED contains both outsourcing encryption and outsourcing decryption functions. However, all of these do not support the attribute update function. In our scheme, we not only support the function of the outsourcing encryption and decryption but also add the function of attribute updates.

7.2. Computing overhead

Our computing cost mainly comes from the private key generation stage of Key Authority, the local data encryption phase of the Data Owner, the outsourcing encryption phase, and the outsourcing decryption phase from the edge node or the proxy, and the local data decryption phase from the Data User.

Table 2
Notations.

Notations	Description
E_1	Exponentiation or multiplication in group G_1
E_2	Exponentiation or multiplication in group G_2
e	Pairing operation in group G_2
S_u	The number of attributes related to the user's private key
c	The number of attributes related to ciphertext
s	The least internal node to satisfy the access structure is satisfied.

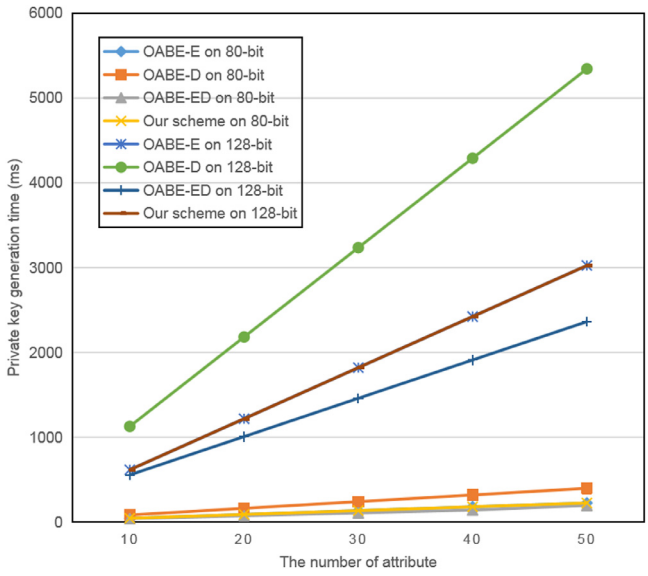


Fig. 3. Private key generation time.

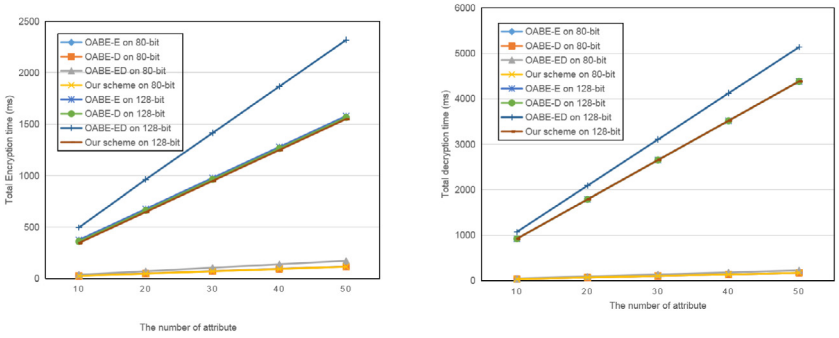
7.2.1. Theoretical calculation loads analysis

For ease of understanding, Table 2 summarizes the symbols used in the following theoretical analysis, and in Table 3, we can see the comparison of our scheme and the calculation cost of schemes by OABE-E, OABE-D, and OABE-ED, in the key generation stages, encryption stages and decryption stages.

In Table 3, we can see that our scheme has nearly half less computing load than OABE-D′s scheme. Our scheme is the same as OABE-E′s scheme, but more $(S_u$- 6) E_1 than OABE-ED′s scheme. This is because in our scheme, to ensure the data security, multiple private keys are generated. In the encryption phase, we outsource part of the encryption to the edge nodes so as to reduce the computing load of resource-limited devices such as DO. The calculation load of OABE-D′s scheme and OABE-ED′s scheme in the local decryption phase is larger than OABE-E′s scheme and our scheme. This is because in OABE-D′s scheme, encryption is not outsourced to edge nodes or the proxy and OABE-ED′s scheme needs to generate all ciphertext, which then the proxy is required to re-encrypt the ciphertext. In the decryption phase, because OABE-E′s scheme does not contain the outsourcing decryption operation, for the calculation load on DU, OABE-E′s scheme is much higher than the other three schemes. In our scheme, e+ $2E_2$ and $2S_u$e+ $2sE_2$ are used to represent the computing load of DU and edge nodes respectively. The computing load of the edge nodes or the proxy in OABE-D′s scheme and OABE-ED′s scheme is higher than that in our scheme, and the computational load of our DU is almost the same. Therefore, in general, our scheme is more efficient than the other three schemes.

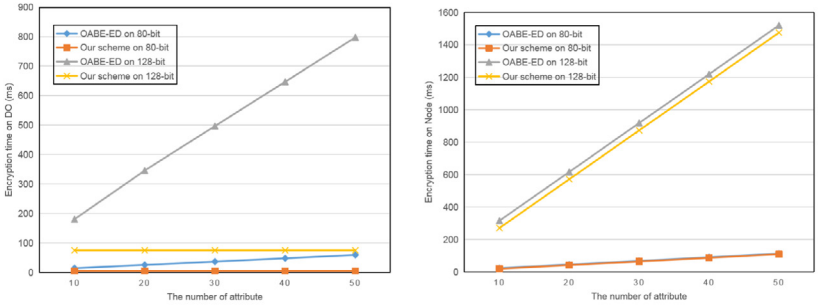
Table 3
The computing overhead comparison of the four schemes.

Scheme	OABE-E	OABE-D	OABE-ED	Our scheme
Key generation	$(1+4S_u)E_1$	$(7S_u+5)E_1$	$(3S_u+7)E_1$	$(1+4S_u)E_1$
Encryption on DO	$2E_2+3E_1$	$2E_2+(2+2c)E_1$	$(c+2)E_1+E_2$	$2E_2+3E_1$
Encryption on Edge Node	$2cE_1$	–	$(2c+1)E_1$	$2(c-1)E_1$
Decryption on DU	$(1+2S_u)e+(2s+2)E_2$	E_1	E_2	$e+2E_2$
Decryption on Edge Node	–	$(2S_u+2)e+2sE_2$	$(2S_u+1)e+(S_u+2s+1)E_2$	$2S_ue+2sE_2$
Key update for updated user	–	–	–	E_1
Key update for non-updated user	–	–	–	E_1
Ciphertext update	–	–	–	E_1



(a) Total encryption time. (b) Total decryption time.

Fig. 4. Total encryption or decryption time.



(a) Encryption time on DO. (b) Encryption time on edge node.

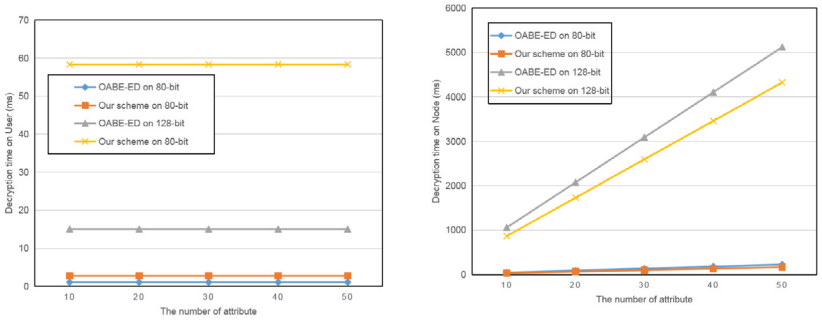
Fig. 5. Encryption time on DO or edge node.

7.2.2. Experimental analysis

The operating system used in the experiments is Ubuntu 18.04.3 with Intel Core i5-7500 CPU@3.40 GHz and 16GB of memory. The GMP library (version 6.1.2) is applied to implement large integer operations. The PBC library (version 0.5.14) is applied to implement pairing calculations and then we chose the type-A curve, with the security level of 80-bit and 128-bit.

Generally, the experimental encryption time is mainly composed of two operations: exponential operation and multiplication operation. Therefore, in our experiment, when the security level is 80 bit and 128 bit respectively, we run each operation 1000 times, and finally, calculate the average value of each operation. For the group G_1, the time of index calculation is 1.125 ms and 15.053 ms respectively. For the group G_2, the time of calculation is 1.124 ms and 15.083 ms respectively. In addition, the time of linear pairing operation is 0.548 ms and 28.192 ms, respectively. Based on these basic calculation time, we compare the performance of our scheme with the other three schemes.

From Fig. 3, no matter under the 80-bit security level or 128-bit security level, we can see that the private key generation calculation time of OABE-D′s scheme is longer than that of our scheme. The calculation time of the private key generation of our scheme is slightly longer than that of OABE-ED′s scheme because in the private key generation stages of our scheme, we have generated multiple private keys to ensure the security of encryption. However, private key generation is calculated on KA without resource limitation. KA is not a limited resource node. It can do some calculations, so it is acceptable. In Fig. 4(a), the total encryption time of four schemes is compared. We can see that only the OABE-ED′s scheme are higher than that of the other three schemes, The total encryption time of our scheme is almost the same as that of OABE-E′s scheme and OABE-D′s scheme.



(a) Decryption time on DU. (b) Decryption time on edge node.

Fig. 6. Decryption time on DU or edge node.

In Fig. 4(b), we compare the total decryption time of the four schemes. Only OABE-D′s scheme is larger than the other schemes, and the total decryption time of our scheme is almost the same as OABE-E′s scheme and OABE-ED′s scheme. In these figures, we can see that the encryption and decryption time increases with the number of attributes. From Fig. 5(a), the calculation time of our scheme is less than OABE-ED′s scheme. We outsource some encryption operations to edge nodes to reduce the computational load of sensors. The data owner's encryption time is a very small constant value under the two bit security levels, 5.623 ms and 75.325 ms respectively, which greatly reduces the computing load of resource-constrained sensors. In Fig. 5(b), we can see that the encryption calculation time of our scheme on the edge node is less than that of OABE-ED′s scheme. The computing load of the edge nodes is reduced. Fig. 6(a) shows the comparison of encryption time between our scheme and OABE-ED′s scheme. Because we need to pair the ciphertext once, our scheme has more matching time than OABE-ED′s scheme. One time pairing time is 5.48 ms and 28.192 ms respectively, and now the data owners are generally mobile phones or computer terminals, which have certain computing power. Therefore, a certain amount of calculation is acceptable to users. In Fig. 6(b), the decryption time of OABE-ED′s scheme at the edge node is longer than that of our scheme. In contrast, our scheme reduces the computing load of the edge nodes.

8. Conclusion

In this paper, we propose an efficient and outsourcing-supported attribute-based access controlling encryption scheme that reduces the computing load in the case of resource-constrained devices. We transfer part of the encryption and decryption load on to the edge nodes, and the update function provides a method for updating the attributes. The proof of security shows that our scheme is secure under the DBDH assumption. Moreover, performance analysis shows that the computing load of our encryption scheme is constantly on the data owner and the data user. Therefore, it solves the problem that a resource-limited device cannot perform a large number of computing operations. At the same time, it shows that our encryption scheme is optimal compared with the other three schemes. In the future, we will consider adding verifiability to the scheme to improve the security of data encryption and consider online and offline methods to improve the efficiency of scheme encryption.

CRediT authorship contribution statement

Hong Zhong: Conceptualization, Methodology, Writing - original draft. **Yiyuan Zhou:** Formal analysis, Software, Validation. **Qingyang Zhang:** Investigation, Visualization. **Yan Xu:** Resources, Data curation. **Jie Cui:** Writing - review & editing, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The work was supported by the National Natural Science Foundation of China (No. U1936220, No. 61872001, No. 61702005), the Special Fund for Key Program of Science and Technology of Anhui Province, China (No. 18030901027), the Open Fund for Discipline Construction, Institute of Physical Science and Information Technology, Anhui University, China. The authors are very grateful to the anonymous referees for their detailed comments and suggestions regarding this paper.

References

[1] A. Zanella, N. Bui, A. Castellani, L. Vangelista, M. Zorzi, Internet of things for smart cities, IEEE Internet Things J. 1 (1) (2014) 22–32.
[2] M. Addington, D. Schodek, Smart Materials and Technologies in Architecture: For the Architecture and Design Professions, Routledge, 2012.
[3] L. Catarinucci, D. De Donno, L. Mainetti, L. Palano, L. Patrono, M.L. Stefanizzi, L. Tarricone, An IoT-aware architecture for smart healthcare systems, IEEE Internet Things J. 2 (6) (2015) 515–526.
[4] H. Demirkan, A smart healthcare systems framework, IT Prof. 15 (5) (2013) 38–45.
[5] D.-M. Han, J.-H. Lim, Smart home energy management system using IEEE 802.15. 4 and zigbee, IEEE Trans. Consum. Electron. 56 (3) (2010) 1403–1410.
[6] N. Kaaniche, M. Laurent, Data security and privacy preservation in cloud storage environments based on cryptographic mechanisms, Comput. Commun. 111 (2017) 120–141.
[7] S. Belguith, N. Kaaniche, A. Jemai, M. Laurent, R. Attia, Pabac: a privacy preserving attribute based framework for fine grained access control in clouds, 2016.
[8] S. Belguith, A. Jemai, R. Attia, Enhancing data security in cloud computing using a lightweight cryptographic algorithm, in: The Eleventh International Conference on Autonomic and Systems, 2015, pp. 98–103.

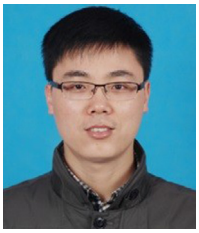
[9] J. Cui, L. Wei, H. Zhong, J. Zhang, Y. Xu, L. Liu, Edge computing in VANETs-an efficient and privacy-preserving cooperative downloading scheme, IEEE J. Sel. Areas Commun. (2020) 1.
[10] J. Zhang, J. Cui, H. Zhong, Z. Chen, L. Liu, PA-CRT: Chinese remainder theorem based conditional privacy-preserving authentication scheme in vehicular Ad-hoc networks, IEEE Trans. Dependable Secure Comput. (2019).
[11] W. Shi, J. Cao, Q. Zhang, Y. Li, L. Xu, Edge computing: Vision and challenges, IEEE Internet Things J. 3 (5) (2016) 637–646.
[12] P. Mach, Z. Becvar, Mobile edge computing: A survey on architecture and computation offloading, IEEE Commun. Surv. Tutor. 19 (3) (2017) 1628–1656.
[13] L. Ren, Q. Zhang, W. Shi, Y. Peng, Edge-based personal computing services: fall detection as a pilot study, Computing 101 (8) (2019) 1436–5057.
[14] A. Sahai, B. Waters, Fuzzy identity-based encryption, in: Annual International Conference on the Theory and Applications of Cryptographic Techniques, Springer, 2005, pp. 457–473.
[15] V. Goyal, O. Pandey, A. Sahai, B. Waters, Attribute-based encryption for fine-grained access control of encrypted data, in: Proceedings of the 13th ACM Conference on Computer and Communications Security, ACM, 2006, pp. 89–98.
[16] Q. Zhang, H. Zhong, J. Cui, L. Ren, W. Shi, AC4AV: A flexible and dynamic access control framework for connected and autonomous vehicles, IEEE Internet Things J. (2020) 1, http://dx.doi.org/10.1109/JIOT.2020.3016961.
[17] X. Yao, Z. Chen, Y. Tian, A lightweight attribute-based encryption scheme for the Internet of Things, Future Gener. Comput. Syst. 49 (2015) 104–112.
[18] W. Shi, X. Zhang, Y. Wang, Q. Zhang, Edge computing: State-of-the-art and future directions, J. Comput. Res. Dev. 56 (1) (2019) 1–21.
[19] M. Green, S. Hohenberger, B. Waters, et al., Outsourcing the decryption of abe ciphertexts, in: USENIX Security Symposium, Vol. 2011, No. 3, 2011.
[20] Q. Huang, Y. Yang, M. Shen, Secure and efficient data collaboration with hierarchical attribute-based encryption in cloud computing, Future Gener. Comput. Syst. 72 (2017) 239–249.
[21] E. Hendrick, B. Schooley, C. Gao, CloudHealth: developing a reliable cloud platform for healthcare applications, in: 2013 IEEE 10th Consumer Communications and Networking Conference, CCNC, IEEE, 2013, pp. 887–891.
[22] O. Gul, M. Al-Qutayri, C.Y. Yeun, Q.H. Vu, Framework of a national level electronic health record system, in: 2012 International Conference on Cloud Computing Technologies, Applications and Management, ICCCTAM, IEEE, 2012, pp. 60–65.
[23] Y. Zhang, M. Qiu, C.-W. Tsai, M.M. Hassan, A. Alamri, Health-CPS: Healthcare cyber-physical system assisted by cloud and big data, IEEE Syst. J. 11 (1) (2015) 88–95.
[24] Q. Zhang, Q. Zhang, W. Shi, H. Zhong, Distributed collaborative execution on the edges and its application to AMBER alerts, IEEE Internet Things J. 5 (5) (2018) 3580–3593, http://dx.doi.org/10.1109/JIOT.2018.2845898.
[25] Q. Zhang, H. Sun, X. Wu, H. Zhong, Edge video analytics for public safety: A review, Proc. IEEE 107 (8) (2019) 1675–1696.
[26] X. Wu, R. Dunne, Z. Yu, W. Shi, STREMS: a smart real-time solution toward enhancing EMS prehospital quality, in: 2017 IEEE/ACM International Conference on Connected Health: Applications, Systems and Engineering Technologies, CHASE, IEEE, 2017, pp. 365–372.
[27] X. Sun, P. Zhang, M. Sookhak, J. Yu, W. Xie, Utilizing fully homomorphic encryption to implement secure medical computation in smart cities, 21 (5) (2017) 831–839.
[28] Z. Cai, H. Yan, P. Li, Z.-a. Huang, C. Gao, Towards secure and flexible EHR sharing in mobile health cloud under static assumptions, Cluster Comput. 20 (3) (2017) 2415–2422.
[29] P. Panchatcharam, S. Vivekanandan, Internet of things (IoT) in healthcare–smart health and surveillance, architectures, security analysis and data transfer: a review, Int. J. Softw. Innov. 7 (2) (2019) 21–40.
[30] Y. Li, G. Wang, L. Nie, Q. Wang, W. Tan, Distance metric optimization driven convolutional neural network for age invariant face recognition, Pattern Recognit. 75 (2018) 51–62.
[31] R.F. Nogueira, R. de Alencar Lotufo, R.C. Machado, Fingerprint liveness detection using convolutional neural networks, IEEE Trans. Inf. Forensics Secur. 11 (6) (2016) 1206–1213.
[32] L. Chen, D.B. Hoang, Novel data protection model in healthcare cloud, in: 2011 IEEE International Conference on High Performance Computing and Communications, IEEE, 2011, pp. 550–555.
[33] J. Bethencourt, A. Sahai, B. Waters, Ciphertext-policy attribute-based encryption, in: 2007 IEEE Symposium on Security and Privacy, SP'07, IEEE, 2007, pp. 321–334.
[34] S. Narayan, M. Gagné, R. Safavi-Naini, Privacy preserving EHR system using attribute-based infrastructure, in: Proceedings of the 2010 ACM Workshop on Cloud Computing Security Workshop, ACM, 2010, pp. 47–52.
[35] N. Attrapadung, B. Libert, E. De Panafieu, Expressive key-policy attribute-based encryption with constant-size ciphertexts, in: International Workshop on Public Key Cryptography, Springer, 2011, pp. 90–108.
[36] B. Waters, Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization, in: International Workshop on Public Key Cryptography, Springer, 2011, pp. 53–70.
[37] S. Dash, S. Biswas, D. Banerjee, A.U. Rahman, Edge and fog computing in healthcare–a review, Scalable Comput.: Pract. Exper. 20 (2) (2019) 191–206.
[38] Y. Rouselakis, B. Waters, Efficient statically-secure large-universe multi-authority attribute-based encryption, in: International Conference on Financial Cryptography and Data Security, Springer, 2015, pp. 315–332.
[39] J. Li, W. Yao, Y. Zhang, H. Qian, J. Han, Flexible and fine-grained attribute-based data storage in cloud computing, IEEE Trans. Serv. Comput. 10 (5) (2016) 785–796.
[40] Z. Liu, Z.L. Jiang, X. Wang, S.-M. Yiu, Practical attribute-based encryption: Outsourcing decryption, attribute revocation and policy updating, J. Netw. Comput. Appl. 108 (2018) 112–123.
[41] J. Li, C. Jia, J. Li, X. Chen, Outsourcing encryption of attribute-based encryption with mapreduce, in: International Conference on Information and Communications Security, Springer, 2012, pp. 191–201.
[42] J. Li, X. Huang, J. Li, X. Chen, Y. Xiang, Securely outsourcing attribute-based encryption with checkability, IEEE Trans. Parallel Distrib. Syst. 25 (8) (2013) 2201–2210.
[43] M. Asim, M. Petkovic, T. Ignatenko, Attribute-based encryption with encryption and decryption outsourcing, in: 12th Australian Information Security Management Conference, ECU Security Research Institute, 2014, pp. 21–28.

Hong Zhong was born in Anhui Province, China, in 1965. She received her Ph.D. degree in computer science from University of Science and Technology of China in 2005. She is currently a professor and Ph.D. supervisor of School of Computer Science and Technology at Anhui University. Her research interests include applied cryptography, IoT security, vehicular ad hoc network, cloud computing security and software-defined networking (SDN). She has over 120 scientific publications in reputable journals (e.g. IEEE Transactions on Dependable and Secure Computing, IEEE Transactions on Parallel and Distributed Systems, IEEE Transactions on Vehicular Technology, IEEE Transactions on Intelligent Transportation Systems, IEEE Transactions on Big Data, IEEE Internet of Things Journal, Information Sciences, Journal of Parallel and Distributed Computing), academic books and international conferences.

Yiyuan Zhou is now a research student in the School of Computer Science and Technology, Anhui University. His research interests are smart healthcare and security.

Qingyang Zhang received the B. Eng. degree in computer science and technology from Anhui University, China in 2014, where he is currently pursuing the Ph.D. candidate. His research interest includes edge computing, computer systems, and security.

Yan Xu is an Associate Professor in the School of Computer Science and Technology, Anhui University. She received Ph.D. degree from University of Science and Technology of China in 2015. Her research interests include network and information security.



Jie Cui was born in Henan Province, China, in 1980. He received his Ph.D. degree in University of Science and Technology of China in 2012. He is currently an associate professor in the School of Computer Science and Technology at Anhui University. His current research interests include applied cryptography, IoT security, vehicular ad hoc network, cloud computing security and software-defined networking (SDN). He has over 80 scientific publications in reputable journals (e.g. IEEE Transactions on Dependable and Secure Computing, IEEE Transactions on Vehicular Technology, IEEE Transactions on Intelligent Transportation Systems, IEEE Transactions on Circuits and Systems, IEEE Internet of Things Journal, Information Sciences, Journal of Parallel and Distributed Computing), academic books and international conferences.