Future Generation Computer Systems 99 (2019) 134–142

Contents lists available at ScienceDirect

Future Generation Computer Systems

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

Efficient and privacy preserving access control scheme for fog-enabled IoT

Kai Fan [a,*], Huiyue Xu [a], Longxiang Gao [b], Hui Li [a], Yintang Yang [c]

[a] State Key Laboratory of Integrated Service Networks, Xidian University, Xi'an, China
[b] School of Information Technology, Deakin University, VIC, Australia
[c] Key Lab. of Minist. of Educ. for Wide Band-Gap Semicon. Materials and Devices, Xidian University, Xi'an, China

HIGHLIGHTS

- We propose an efficient and privacy preserving access control scheme for fog-enabled IoT.
- We design the efficient and verifiable outsourced decryption assisted by fog nodes in the scheme.
- We propose a secure approach to transform attributes to be anonymous and authenticable.
- The performance simulation show that proposed scheme is secure and highly efficient.

ARTICLE INFO

Article history:
Received 30 September 2018
Received in revised form 14 March 2019
Accepted 1 April 2019
Available online 17 April 2019

Keywords:
Cloud computing
Fog computing
Access control
Attribute-based encryption
Privacy preserving
Decryption outsourcing

ABSTRACT

The fog-to-things paradigm is introduced to mitigate the heavy burden on the edge of cloud-based network due to the centralized processing and storing of the massive volume of IoT data. Fog-enabled IoT architectures ensure small latency and enough computing resource that enables real time devices and applications. However, there still exist security and privacy challenges on data access control for fog-enabled IoT. Ciphertext-policy attribute-based encryption (CP-ABE) can be adopted to realize data access control in cloud-fog computing systems. In this paper, we propose an efficient and privacy preserving outsourced multi-authority access control scheme, named PPO-MACS. All attributes of users are transformed to be anonymous and authenticable to realize privacy preserving. And the verifiable outsourced decryption is introduced to reduce computation overheads on the end user side. Meanwhile, an efficient user revocation method is proposed. Security and performance analysis show that our scheme is secure and highly efficient.

1. Introduction

The Internet of Things (IoT) is the latest evolution of the internet, including an great deal of connected physical devices and applications [1]. While IoT devices have limited computing resources and low storage, it can gain advantage from the powerful capabilities and resources of the cloud to deal with its technological constraints. With substantially unlimited computing resources, low-cost, on-demand storage capacity, and available resources from anywhere, the cloud is a cost-effective and convenient solution to offset IoT technological constraints [2–4]. Cloud-enabled IoT has drawn considerable attention due to its great potential. However, it suffers from various drawbacks such as huge network latency due to the centralized processing and the massive volume of data [5].

Therefore, fog computing [6–8] located between IoT devices and the cloud infrastructure is introduced to extend computing, storage and network services to the edge of the internet. Fog computing nodes are located between the traditional centralized cloud and the IoT devices. Therefore, resources and services of the fog nodes are available and are closer to the end devices. And the network delays induced by deployments of cloud infrastructure can be reduced. Thus, fog-enabled IoT architectures ensure small latency and enough computing resource that enables real time devices and applications. However, it is also confronted with various issues of privacy preserving and efficiency of the data access control, which raise serious concerns [9–11].

To address such issues in fog-enabled IoT, it is urgently needed to adopt an efficient access control scheme supporting privacy preserving. Ciphertext-policy attribute-based encryption [12] (CP-ABE) is consider as the most suitable technology for providing IoT with data confidentiality and fine-grained data access control. It enables data publishers to define flexible access policy. However, the access policy sent along with a ciphertext explicitly

* Corresponding author.
E-mail addresses: kfan@mail.xidian.edu.cn (K. Fan), huiyuexu@stu.xidian.edu.cn (H. Xu), longxiang.gao@deakin.edu.au (L. Gao), lihui@mail.xidian.edu.cn (H. Li), ytyang@xidian.edu.cn (Y. Yang).

https://doi.org/10.1016/j.future.2019.04.003
0167-739X/

may leak sensitive information of data owners and end device users. And the processes of encryption and decryption in CP-ABE systems are time-consuming. The computation overheads on data users is too heavy. Therefore, an efficient and privacy preserving multi-authority CP-ABE access control scheme is needed to provide a solution supporting preserve privacy in fog-enabled IoT.

1.1. Related work

In 2007, Bethencourt et al. [12] put forward the first CP-ABE scheme. Over the last decade, many CP-ABE schemes [13–16] are proposed. However, most of them does not support access privacy preserving and are lack of efficiency.

1.1.1. Outsourced computation

Note that most of existing scheme are time-consuming and lack of efficiency. To improve the efficiency and reduce the overheads of users, Green et al. [17] proposed an ABE scheme supporting outsourced computation in 2011. Then several schemes [18, 19] supporting outsourced computation are proposed. In these schemes, the private keys are divided into user keys and transformation keys. Thus, complex decryption computations are outsourced to the cloud server, and users only need one exponentiation operation to recover the plaintext. However, most of them did not consider correctness of the partial decryption result. In 2013, Lai et al. [20] proposed an ABE scheme supporting verifiable outsourced decryption. In 2016, Mao et al. [21] proposed a generic construct of ABE with verifiable outsourced decryption, which greatly reduce the computation cost of users. In 2017, Fan et al. [22] proposed a verifiable outsourced multi-authority access control scheme in fog-cloud computing. This scheme provided a verification method of outsourced encryption and decryption. In 2018, Zuo et al. [23] proposed an ABE scheme supporting outsourced decryption for fog computing, which is concrete CCA-secure. In 2019, Li et al. [24] proposed a scheme supporting verifiable outsourced encryption and decryption. In this scheme, they proposed a secure method to generate the transformation key. However, most of mentioned schemes with outsourced decryption did not consider access policy privacy preserving.

1.1.2. Policy preserving

To address the problem of access policy leaking, several schemes supporting privacy preserving were proposed. Nishide et al. [25] proposed a partially access policy hidden scheme and then the schemes [26,27] were proposed, while all of them were based on simple and less expressive 'AND' gate access structure. In 2015, Xu et al. [28] and Qian et al. [29] proposed policy hidden schemes which adopted tree access structure to improve expressiveness of access policy. In 2011, Waters [30] firstly introduced the Linear Secret Sharing Schemes (LSSS) matrix as the access structure, which is a more expressive. In 2012, Lai et al. [31] firstly proposed a scheme based on LSSS which only supported partial policy hidden but did not support user revocation. While most of the schemes mentioned above are based on the single-authority. While the multi-authority CP-ABE schemes are more suitable for IoT access control. In 2011, Lewko and Waters [14] presented a decentralizing multi-authority ABE scheme which was collusion resistant. Then some schemes [22,32,33]with multiple authorities were proposed. In 2018, Zhong et al. [34] proposed a multi-authority access control scheme based on LSSS with policy hidden. This scheme introduced the one-way anonymous key agreement protocol [35] to realize policy hidden. However, the computation cost of this scheme was extremely expensive, which does not satisfy the real-time operation requirements of IoT.

1.2. Our contribution

According to the security and privacy challenges in data access control for IoT devices, we construct an efficient and privacy preserving outsourced multi-authority access control scheme for fog-enabled IoT, named PPO-MACS. The main contributions are as follows:

(1) We propose a privacy preserving outsourced multi-authority access control scheme, which supports access policy hidden, collusion assistance and efficient user revocation.

(2) The verifiable outsourced decryption assisted by fog nodes is adopted in the paper, which greatly reduces computation cost on the end user side.

(3) We design a secure algorithm to preserving access policy. In this paper, all attributes are transformed to be anonymous. They are sent to cloud and can be securely authenticated by decryption proxy server. In particular, all operation of transformation and authentication are executed on fog node with enough computation capacity instead of end devices.

(4) We provide a security and performance analysis of our scheme. The result shows that our scheme is secure in term of privacy preserving and highly efficient in term of decryption computing.

1.3. Organization

The rest of the paper is organized as follows: Section 2 give the preliminaries of the proposed scheme. In Section 3, we present the system model and the security assumptions. Section 4 describes the construction of our scheme in detail. Then, the security analysis is provided in Section 5 and performance analysis is provided in Section 6. Finally, Section 7 concludes the paper.

2. Preliminaries

In this section, the formal definition of bilinear maps is first given. Then we describe relevant information of access structures and Linear Secret Sharing Schemes (LSSS).

2.1. Bilinear maps

Definition 1 (*Bilinear Maps*)**.** Let $\mathbb{G}$ and $\mathbb{G}_T$ be two multiplicative cyclic groups of prime order p. A bilinear map is a map $e: \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ with the following properties

(1) Bilinearity: For all $u, v \in \mathbb{G}$ and $a, b \in \mathbb{Z}_p$, we have $e(u^a, v^b) = e(u, v)^{ab}$.

(2) Non-degeneracy $e(g, g) \neq 1$.

(3) Computability: There is an efficient algorithm to compute $e(g, g)$ for any $g \in \mathbb{G}$.

Notice that the map e is symmetric since $e(g^a, g^b) = e(g, g)^{ab} = e(g^b, g^a)$.

2.2. Access structure

Definition 2 (*Access Structure [35]*)**.** Let $\{P_1, P_2, \ldots, P_n\}$ be a set of parties. A collection $\mathbb{T} \subseteq 2^{\{P_1, P_2, \ldots, P_n\}}$ is monotone for $\forall B, C$: if $B \in \mathbb{T}$ and $B \subseteq C$ then $C \in \mathbb{T}$. An access structure (respectively, monotone access structure) is a collection (respectively, monotone collection) $\mathbb{T}$ of non-empty subsets of $\{P_1, P_2, \ldots, P_n\}$, i.e., $\mathbb{T} \subseteq 2^{\{P_1, P_2, \ldots, P_n\}} \backslash \{\phi\}$. The sets in $\mathbb{T}$ are called the authorized sets, and the sets not in $\mathbb{A}$ are called the unauthorized sets.

2.3. Linear secret sharing schemes

Definition 3 (*Linear Secret Sharing Schemes (LSSS [35])*). Let $\prod$ be a secret-sharing scheme over a set of parties $\mathcal{P}$ with realizing an access structure $\mathbb{A}$. It is called linear secret sharing scheme over $\mathbb{Z}_p$ if

(1) The shares for each party form a vector over $\mathbb{Z}_p$.

(2) There exists a $l \times n$ matrix $\mathbb{A}$ called the share-generating matrix for $\prod$. For all $i = 1, 2, \ldots, l$, the function ρ defines the party labeling $i'th$ row of $\mathbb{A}$ as $\rho(i)$. Suppose there is a vector $\overrightarrow{v}^T = (s, r_2, \ldots, r_n)$, where $s \in \mathbb{Z}_p$ is the secret to be shared, and $r_2, \ldots, r_n \in \mathbb{Z}_p$ are randomly selected. $\mathbb{A}\overrightarrow{v}$ is the vector of l shares of the secret s according to Π. The share $(\mathbb{A}\overrightarrow{v})_i$ belongs to party $\rho(i)$

Linear Reconstruction [35]: Let $\prod$ be a LSSS for the access structure $\mathbb{T}$. Let $S \in \mathbb{T}$ be any authorized set, and $I \subset \{1, 2, \ldots, l\}$ be defined as $I \subset \{i: \rho(i) \in S\}$. There exists constants $\{w_i \in \mathbb{Z}_p\}_{i \in I}$ such that, if $\{\lambda_i\}$ are valid shares of any secret s according to Π, then $\sum_{i \in I} w_i \lambda = s$. These constants $\{w_i\}$ can be found in time polynomial in the size of the share-generating matrix $\mathbb{A}$.

3. System model

In this section, we briefly introduce the system model, security assumptions and framework of our scheme.

3.1. System model

As given in Fig. 1, we consider a simple three level hierarchy in our system of cloud-fog assisted IoT. In this system, all IoT devices is directly connected to fog nodes instead of the cloud. The proxy servers are deployed on the fog nodes which mitigate the heavy burden on the IoT devices and storing of the massive IoT data. And fog nodes are connected to cloud and managed by the cloud.

As shown in Fig. 2, system model of our scheme has seven entities: a cloud service provider (CSP), fog nodes (FNs), proxy server (PS), a global certificate authority (CA), attribute authorities (AAs), data owners (DOs) and end users (EUs). The CA and the AAs are deployed in special severs for authorities, which are independent of the cloud and the fog node. They provide the efficient and fine-grained access control for cloud-fog computing system.

The CA is the global certificate authority in the system. It is responsible for the registration of AAs and EUs and issuing the unique user and authority identifier. It does not participate in any keys and attributes management and is fully trusted.

AAs are independent to each other. They are responsible for transforming attributes within their domain to be anonymous and issuing them to relevant end users. Each AA is also in charge of the key generation and publishing within its domain. In this way, we avoid placing absolute trust in a single central authority which must remain active and uncorrupted throughout the lifetime. So we avoid the risk of single point of failure incurred by relying on a central authority.

The CSP is in charge of storing the massive decrypted data, the anonymous attributes list and the proxy keys list belonging to end users. It also handles data access request from users and performs user attribute and key update operation for the revoked user.

DOs are responsible for the definition of access policy and the data encryption according to the policy. Then the decrypted data is uploaded to the CSP.

FNs are near-user devices to carry out a substantial amount of storage, communication and computation. PSs are deployed on FNs to mitigate the heavy burden of the end users. They are in charge of data transmission, user attribute authentication and outsourced decryption.

EUs can obtain their secret keys from the relevant authorities. After submitting data access request to the CSP and asking PS to decrypt the ciphertext, they download the partially decrypted ciphertext from PS and can recover it correctly with user secret key.

In special, an attribute transform algorithm only run by AAs and DOs is introduced in our scheme to transform attribute to be anonymous.

3.2. Security assumptions

We consider a common and applicable system model in the paper. To deal with realistic security threats, we provides security assumptions as follows to simulate the common security environment of fog-enabled IoT. These assumptions will make our system meet the requirements of common multi-authority attribute-based encryption access control system model.

(1) The CA is fully trusted in the system, which means it would not leak the data in the CA and collude with any AA or users.

(2) Each AA are trusted, but can be corrupted by the adversary.

(3) The CSP and each PS on the FN are semi-trusted. They are honest-but-curious. They are "honest" and will ensure the security of data and perform the assigned tasks, but they are also "curious" to infer and analyze the data so as to obtain privacy information.

(4) EUs are dishonest and may collude to get unauthorized data. We suppose the revoked EU cannot ask keys from AA.

3.3. System framework

Definition 4 (*PPO-MACS: Privacy Preserving Outsourced Multi-Authority Access Control Scheme in Fog-Enabled IoT*). *GlobalSetup*$(\lambda) \rightarrow \{GP, aid, uid\}$: The global setup algorithm is run by CA. It takes in a security parameter λ and outputs global parameters GP, the end user identifier uid and the authority identifier aid.

AuthSetup$(GP, aid) \rightarrow \{ATK_{aid}, SK_{aid}, PK_{aid}\}$: The authority setup is run by AA. It takes as input the GP and the aid assigned by CA. Its outputs are the attribute transform key ATK_{aid}, the public key PK_{aid}, and the secret key SK_{aid}.

AttrTrans$(GP, ATK_{aid}, S) \rightarrow S'$: The user attribute transform algorithm is run by AA and DO. On input of GP, ATK_{aid} and the attribute set S, it outputs transformed attribute set S'.

Encrypt$(GP, M, T, PK_{aid}) \rightarrow CT$: The encryption algorithm is run by DO. It takes in the GP, a message M, access policy T and the public keys $\{PK_{aid}\}$ of relevant authorities. Its output is the decrypted ciphertext CT.

KeyGen$(GP, uid, SK_{aid}, S) \rightarrow \{UAK_{x,uid}, PrxK_{\text{aid},uid}, USK_{uid}\}$: The key generation algorithm is run by AA. It takes as input the GP, a uid, a secret key SK_{aid} and an attribute set S belonging to an end user EU_{uid}. Then it outputs the user authenticate key $UAK_{x,uid}$, the proxy key $PrxK_{\text{aid},uid}$ and the user secret key USK_{uid}.

UserAuthen$(GP, CT, uid, UAK_{x,uid}) \rightarrow v_{uid}$: The PS on FN runs the user authenticate algorithm. It takes in the GP, a ciphertext CT, a uid and a user authenticate key UAK_{uid}. Then it generates the authenticate result v_{uid} as output.

ProxyDec$(GP, CT, v_{user}, PrxK_{\text{aid},uid}) \rightarrow CT'$: The PS runs the proxy decryption algorithm. It takes in the GP, a ciphertext CT, the authenticate result v_{uid} and a proxy key $PrxK_{\text{aid},uid}$. Then it outputs the partially decrypted ciphertext CT'.

UserDec$(GP, CT, CT', USK_{uid}) \rightarrow M$: The end user EU runs the user decryption algorithm. On the input of the GP, a ciphertext CT, a partially decrypted ciphertext CT', a user secret key USK_{uid}, it outputs the original plain message M.

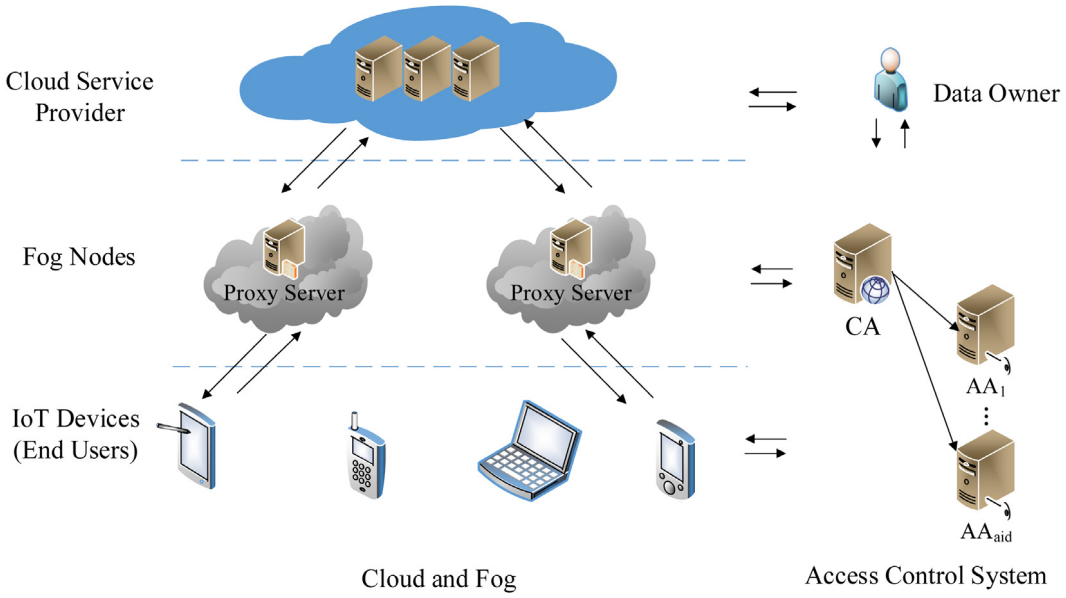


Fig. 1. System model of cloud-fog computing for IoT.

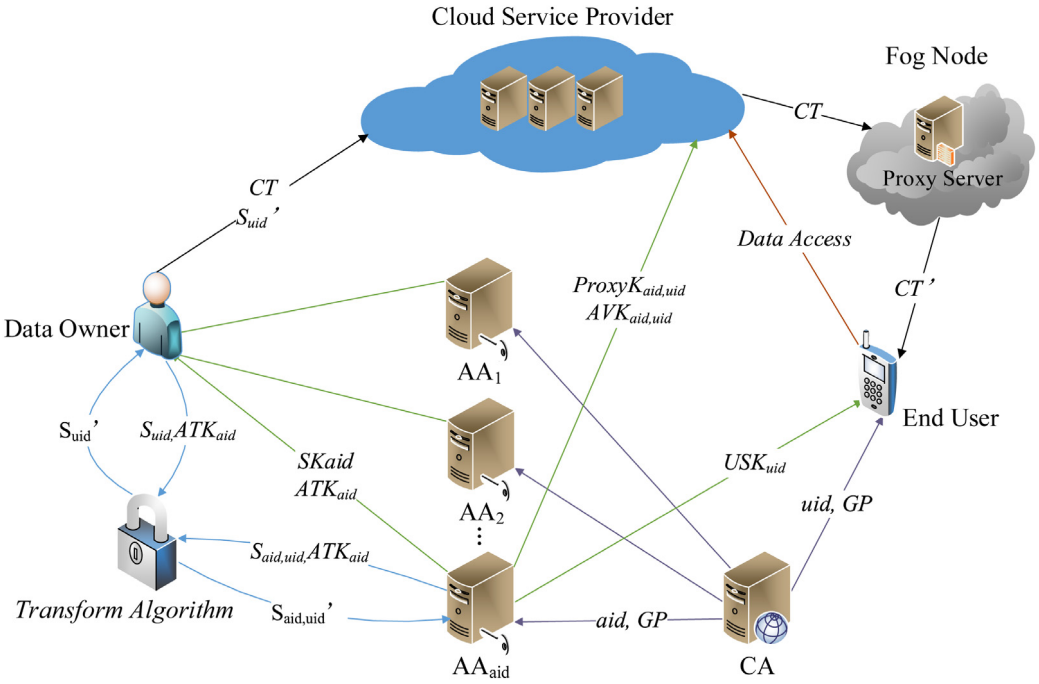


Fig. 2. System model of PPO-MACS.

ProxyVerify$(CT, CT') \rightarrow v_{proxy}$: The proxy verify algorithm is run by the end user. On input of the verify ciphertext C_v in CT and the partially decrypted ciphertext CT', it outputs the verification result.

UserRevoke$(L_K, L_{S_{attr}}, uid) \rightarrow (L_K{}', L_{S_{attr}}{}')$: The user revocation algorithm is run by the CSP. It takes in a *uid* and the list $(L_K, L_{S_{attr}})$ including the proxy key and attribute set which is related to *uid*. Then it outputs the updated list $(L_K{}', L_{S_{attr}}{}')$.

4. System construction

In this section, we first give the detailed construction of our multi-authority CP-ABE access control scheme. Then we give the outsourcing verification method. Finally, we present a user revocation method for our scheme.

4.1. Overview

Our construction is based on [14]. In our approach, all attributes of end users are transformed to be anonymous to preserve policy privacy of ciphertext. We adopt the one-way anonymous key agreement protocol [36] to anonymize each attribute x as $x' = e((g^{\varphi_{aid}})^{\psi}, H(x))$. And they can be securely authenticated before the decryption phase. In addition, we adopt a method of outsourced decryption to reduce computation overhead on the end user side. Complex decryption computations are outsourced to the proxy servers, and end users only need one exponentiation operation to recover the plaintext. Finally, we give an efficient and secure method of user revocation to ensure flexibility of the system.

4.2. Construction of PPO-MACS

(1) System Setup

GlobalSetup(λ) $\rightarrow$ $\{GP, aid, uid\}$: The global setup algorithm takes in a security parameter λ. It chooses two multiplicative groups $\mathbb{G}$ and $\mathbb{G}_T$ of the prime order p. Let g be the generator of $\mathbb{G}$ and e: $\mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ be a bilinear map. Then, the algorithm adopts a collision-resistant hash function H: $\{0, 1\}^* \rightarrow \mathbb{G}_T$ which maps global identities to an element of $\mathbb{G}_T$ and a hash function H': $\mathbb{G}_T \rightarrow \mathbb{Z}_p$ which maps attributes to an element in $\mathbb{G}_T$. After that, each authority and user register itself to the global authority. A unique global identifier *aid* to each legitimate authorities and a unique global user identifier *uid* to each authorized user are assigned by CA. Finally, CA chooses a random number $\eta \in \mathbb{Z}_p$ and publish global parameters as

$$GP = \{p, \mathbb{G}, \mathbb{G}_T, e, g, g^{\eta}, H, H'\}. \tag{1}$$

AuthSetup(GP, aid) $\rightarrow$ $\{ATK_{aid}, SK_{aid}, PK_{aid}\}$: Let S_{aid} denote attributes set managed by the AA_{aid}, where $S_{aid1} \cap S_{aid2} = \varnothing (aid1 \neq aid2')$. The authority setup algorithm first chooses random number $\alpha_{aid} \in \mathbb{Z}_p$. For each attribute x in S_{aid}, AA_{aid} chooses random numbers $\beta_x, h_x, \gamma_x \in \mathbb{Z}_p$ and publishes master secret key as $SK_{aid} = \{\alpha_{aid}, \beta_x, h_x, \gamma_x\}$ and public key as $PK_{aid} = \{e(g, g)^{\alpha_{aid}}, e(g, g)^{\beta_x}, g^{h_x}\}$. It then chooses random numbers $\varphi_{aid}, \psi \in \mathbb{Z}_p$ and generates the attribute transform key as $ATK_{aid} = \{g^{\varphi_{aid}}, \psi\}$ for each attribute authority, which used in attribute transform algorithm.

(2) Attribute Transformation

AttrTrans(GP, ATK_{aid}, S) $\rightarrow$ S': The user attribute transform algorithm takes the global parameters, an attribute transform key and an attribute set as input. For each attribute $x \in S$, the algorithm generates the transformed attribute set as

$$S' = \{x' | x' = e((g^{\varphi_{aid}})^{\psi}, H(x)) = e(g^{\varphi_{aid}}, H(x)^{\psi})\} \tag{2}$$

and outputs it.

(3) Key Generation

KeyGen(GP, uid, SK_{aid}, S) $\rightarrow$ $\{UAK_{uid}, PrxK_{\text{aid},uid}, USK_{uid}\}$: As an end user EU_{uid} wants to access the data, it should obtain its secret keys from the relevant AAs.

Firstly, the authority AA_{aid} assigns the attributes set $S_{aid,uid} \subseteq S$ to the user EU_{uid}. For each attribute $x \in S_{aid,uid}$, AA_{aid} runs the attribute transform algorithm to generate the set $S_{aid,uid}'$ and send it to the CSP. In the CSP, the set will be added into the user attribute set list $L_{S_{uid}}$ as $L_{S_{uid}} = L_{S_{uid}} \cup S_{uid}'$. The $L_{S_{uid}}$ is assigned to an end user EU_{uid} by the CSP and store attributes belonging to EU_{uid}.

Then, AA_{aid} chooses a random number $\sigma_{uid} \in \mathbb{Z}_p$ for each user and issues the user secret keys as $USK_{uid} = \{\sigma_{uid}\}$. The user secret keys then will be sent to each end user EU_{uid}. For each attribute $x \in S_{aid,uid}'$, AA_{aid} generates a user attribute authenticate key as $UAK_{x,uid} = g^{\beta_x} H(uid)^{\gamma_x}$ associated with *uid*. It also computes $P_{x,uid} = g^{h_x/\sigma_{uid}}$, $K_{\text{aid},uid} = (g^{\alpha_{aid}} g^{\eta})^{1/\sigma_{uid}}$, $L_{aid} = g^{1/\sigma_{uid}}$ and publishes the proxy key as:

$$ProxyK_{\text{aid},uid} = \{K_{\text{aid},uid}, L_{aid}, P_{x,uid}\}. \tag{3}$$

The keys set $\{ProxyK_{aid,uid}, UAK_{uid}\}$ are sent to CSP who will add them in its key list L_K as $L_K = L_K \cup \{uid, ProxyK_{aid,uid}, UAK_{uid}\}$.

(4) Data Encryption

Encrypt($GP, M, \mathbb{T}, PK_{aid}$) $\rightarrow$ CT: Let S_T denote the attributes set of access policy$\mathbb{T}$. For each attribute $x \in S_T$, DO runs *AttrTrans* algorithm to generate S_T' so that the attributes in access policy is anonymous. Then, it generate the $l \times n$ LSSS matrix $\mathbb{A}$ with ρ mapping its rows to each $x \in S_T'$, where n is the number of attributes in S_T'. Then, DO chooses a random secret exponent $s \in \mathbb{Z}_p$ and a random $v = (s, s_2, s_3, \ldots, s_l) \in \mathbb{Z}_p^l$, where $s_2, s_3, \ldots, s_l$ is used to share the secret s. Let λ_i denote $\mathbb{A}_i v$, where $\mathbb{A}_i$ is row i of $\mathbb{A}$. It also chooses a random vector $\omega = (0, t_1, t_2, \ldots, t_{l-1})^T \in \mathbb{Z}_p^l$ and let ω_i denote $A_i v$. For $\forall i \in \{1, \ldots, n\}$, DO randomly chosen $\mu_i \in \mathbb{Z}_p$ and computes CT as:

$$CT = \begin{Bmatrix} (A, \rho), C_0 = Me(g, g)^{\sum_{aid} \alpha_{aid} s}, C_v = H'(e(g, g)^{\sum_{aid} \alpha_{aid} s}) \\ C' = g^s, C_{1,i} = g^{\eta \lambda_i} g^{-h_{\rho(i)} \mu_i}, C_{2,i} = g^{\mu_i}, \\ D_{1,i} = e(g, g)^{\beta_{\rho(i)} \mu_i}, D_{2,i} = g^{\gamma_{\rho(i)} \mu_i} g^{\omega_i} \end{Bmatrix} \tag{4}$$

and it then will be sent to CSP. Note that C', $C_{1,i}$, $C_{2,i}$ are designed as attribute ciphertext, $D_{1,i}$, $D_{2,i}$ are designed as attribute authenticate ciphertext and C_v is designed to verify the correctness of the partially decrypted ciphertext.

(5) Data Decryption

UserAuthen($GP, CT, uid, UAK_{x,uid}$) $\rightarrow$ v_{uid}: When an end user with identifier *uid* submits a data access request, CSP will first read the relevant anonymous attribute set S_{uid} from the list $L_{S_{uid}}$ and then send it to the relevant PS. The set S_{uid} can be authenticated by PS to avoid mismatching of *uid* and attributes. The PS then chooses constants $c_i \in \mathbb{Z}_p^*$ so as to $\sum_x c_i A_i = (1, 0, \ldots, 0)$. It then computes:

$$v_{user} = \prod_i \left(\frac{D_{1,i} e(H(uid), D_{2,i})}{e(UAK_{\rho(i),uid}, C_{2,i})}\right)^{c_i} = \prod_i e(H(uid), g)^{\omega_i c_i}. \tag{5}$$

If $v_{user} = 1$, each attribute $x \in S_{uid}$ is matches with*uid* and PS can run the proxy decryption algorithm. Otherwise $v_{user} \neq 1$, PS outputs $\perp$ and halts decryption immediately.

ProxyDec($GP, v_{user}, CT, PrxK_{\text{aid},uid}$) $\rightarrow$ CT': On the condition that $v_{user} = 1$, PS chooses constants $\widetilde{\omega}_i \in \mathbb{Z}_p^*$ such that if $\{\lambda_i\}$ are valid shares of s according M, then $\sum_i \widetilde{\omega}_i \lambda_i = s$, where $\forall i \in \{0, 1, 2, \ldots, n\}$. It then computes:

$$CT_{aid} = \frac{e(C', K_{\text{aid},uid})}{\prod_i (e(C_{1,i}, L_{aid}) e(C_{2,i}, P_{\rho(i),uid}))^{\widetilde{\omega}_i}} = e(g, g)^{s\alpha_{aid}/\sigma_{uid}}. \tag{6}$$

Then, the partially decrypted ciphertext is generated as: $CT' = \prod_{aid} e(g, g)^{s\alpha_{aid}/\sigma_{uid}} = e(g, g)^{\sum_{aid} \alpha_{aid} s/\sigma_{uid}}$, which will be sent to the end user EU_{uid}.

UserDec(GP, CT, CT', USK) $\rightarrow$ M: Let $S_{uid} = \sum_{aid} S_{aid,uid}$. EU_{uid} computes $CT'' = CT'^{\sigma_{uid}}$ and verifies correctness of proxy decryption though proxy verify algorithm. If CT'' verified successfully, EU recovers message as

$$M = C_0 / CT''. \tag{7}$$

Otherwise, the algorithm halts.

4.3. Outsourcing verification

ProxyVerify(CT, CT') $\rightarrow$ v_{proxy}: If fog nodes are attacked, EU will receive the incorrect partially decrypted ciphertext and cannot recover message.

On input of the verify ciphertext C_v in CT and the partially decrypted ciphertext CT', EU computes $v = H(CT')$. If $v = C_v$, then we have $v_{proxy} = 1$, which means the verification is successful. Otherwise let $v_{proxy} = 0$, which means CT' is incorrect.

4.4. User revocation scheme

UserRevoke($L_K, L_{S_{attr}}, uid$) $\rightarrow$ ($L_K', L_{S_{attr}}'$): When CSP receive a revocation signal, the keys $\{PrxK_{aid,uid}, UAK_{uid}\}$ and the attributes set S_{uid} will be deleted from ($L_K, L_{S_{attr}}$) stored on the CSP. Then the updated list ($L_K', L_{S_{attr}}'$) is stored on the CSP. Without the deleted data, PS cannot run the *ProxyDec* algorithm for the revoked EU_{uid}. So that EU_{uid} cannot obtain the correct C' to recover M. Updating the keys and re-encrypting the ciphertext are not needed in our scheme.

5. Security analysis

In this section, the comprehensive security analysis of PPO-MACS is provided.

5.1. Correctness

The correctness of our scheme can be proved by the equations as following.

5.1.1. Data decryption

For $\forall i \in \{0, 1, 2, \ldots, n\}$, we recall that $\sum_i \widetilde{\omega}_i \lambda_i = s$. Then we have:

$$\begin{aligned} CT_{aid} &= \frac{e(C', K_{\text{aid},uid})}{\prod_i (e(C_{1,i}, L_{aid}) e(C_{2,i}, P_{\rho(i),uid}))^{\omega_i}} \\ &= \frac{e(g^s, g^{\alpha_{aid}/\sigma_{uid}} g^{\eta/\sigma_{uid}})}{\prod_i (e(g^{\eta\lambda_i} g^{-h_{\rho(i)}\mu_i}, g^{1/\sigma_{uid}}) e(g^{\mu_i}, g^{h_{\rho(i)}/\sigma_{uid}}))^{\omega_i}} \\ &= \frac{e(g,g)^{s\alpha_{aid}/\sigma_{uid}} e(g,g)^{s\eta/\sigma_{uid}}}{\prod_i (e(g,g)^{\eta\lambda_i})^{\omega_i/\sigma_{uid}}} = e(g,g)^{s\alpha_{sid}/\sigma_{uid}} \end{aligned} \tag{8}$$

Therefore:

$$\frac{C_0}{CT'^{\sigma_{uid}}} = \frac{C_0}{(\prod_{aid} CT_{aid})^{\sigma_{aid}}} = \frac{M e(g,g)^{\sum_{aid} \alpha_{aid} s}}{e(g,g)^{\sum_{aid} \alpha_{aid} s}} = M. \tag{9}$$

Thus, data decryption algorithm satisfies correctness.

5.1.2. Attribute authentication

We recall that $\omega_i = A_i \omega$, where $\omega \cdot (1, 0, \ldots, 0) = 0$. If the each attribute $x \in S_{uid}$matches with*uid*, then we have:

$$\begin{aligned} v_{user} &= \prod_i \left(\frac{D_{1,i} e(H(uid), D_{2,i})}{e(UAK_{\rho(i),uid}, C_{2,i})} \right)^{c_i} \\ &= \prod_i \left(\frac{e(g,g)^{\beta_{\rho(i)}\mu_i} e(H(uid), g)^{\gamma_{\rho(i)}\mu_i} e(H(uid), g)^{\omega_i}}{e(g,g)^{\beta_{\rho(i)}\mu_i} e(H(uid), g)^{\gamma_{\rho(i)}\mu_i}} \right)^{c_i} \\ &= \prod_i e(H(uid), g)^{\omega_i c_i} = 1. \end{aligned} \tag{10}$$

Thus, attribute authentication algorithm satisfies correctness.

5.2. Policy preserving

5.2.1. Attribute anonymity

Before DO encrypts data on the policy and AA issues attributes to EU. Each attribute is transformed to remain anonymous to the CSP and FNs and is securely stored in the CSP. In our scheme, we adopt the one-way anonymous key agreement protocol [36] to anonymize each attribute x as $x' = e((g^{\varphi_{aid}})^{\psi}, H(x))$. Only AAs and the authorized DOs has $H(x)^{\varphi_{aid}}$ to compute x', while without $H(x)^{\varphi_{aid}}$ anyone cannot guess x from x' because the random exponent ψ. Thus, users cannot analyze and infer the access policy as they cannot get any attribute information.

5.2.2. Attribute authentication

Anonymous attributes associated with *uid* only can be authenticated by PS with the right *uid* and the proxy decryption phase can be perform only for authenticated users. We introduce redundant components $D_{1,i}$ and $D_{2,i}$ in the ciphertext and $H(uid)$ in the attribute authenticate key. In the attribute authenticate algorithm, we compute v_{user} containing $e(H(uid), g)^{\omega_i}$, where ω_i is a share of 0. Only the matched *uid* can make that $e(H(uid), g)$ terms cancel out such that we have $v_{user} = 1$.

Table 2
Complexity comparison.

Scheme	*ProxyDec Complexity*	*UserDec Complexity*
Lewko's [14]	/	$lE_{G_T} + 2mlP$
Zhong's [34]	/	$mlE_{G_T} + 2lmP$
Our scheme	$mlE_{G_T} + (2l+1)mP$	$E_{G_T} + O(H')$

5.3. Data security

5.3.1. Data confidentiality

Proxy decryption only can be performed by users who are authenticated and whose attributes satisfying the access policy. The CSP and PSs cannot get user secret key so they cannot completely decrypt ciphertext to recover message. In addition, the revoked users cannot obtain the correct partially decrypted ciphertext because their proxy keys have been deleted. Therefore our scheme satisfies the data confidentiality.

5.3.2. Collusion resistance

The proxy keys issued by different AA are associated with*uid* and specific attributes. Users with different proxy keys cannot decrypt the ciphertext. In addition, users with deferent *uid* cannot collude to be authenticated, so they cannot ask proxy servers decrypting ciphertext. Therefore our scheme is collusion resistant.

6. Performance analysis

In this section, the performance analysis is provided as follows.

6.1. Supported functions comparison

As shown in Table 1, we compare our scheme with existing multi-authority CP-ABE access control schemes. It shows that our proposal is more flexible than other schemes and supports abundant functions, which is more suitable for IoT than the previous relevant schemes.

6.2. Computation complexity comparison

Let P be the time of pairing function, E_G and E_{G_T} be the time of exponentiation in G and G_T, $O(H')$ be the time of hash function H'. We recall that l is the number of attributes in the access policy and m is the number of authorities. Table 2 shows the complexity comparison with related schemes. It shows that our scheme has less computation complexity on the user side than the relevant schemes because of most of decryption computation are mitigated to fog nodes.

6.3. Performance simulation

The scheme is implemented in Charm [37], an extensible python-based framework for the rapidly prototyping of cryptographic schemes. Charm allows us to write code similar to the theoretical implementations. Charm also provides routines for applying and using LSSS schemes needed for attribute-based encryption systems.

(a) Outsourced decryption time

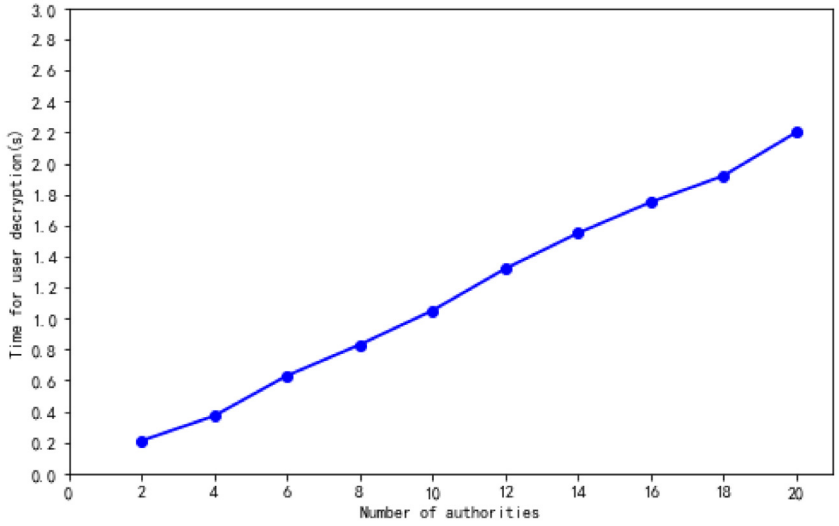
(b) User decryption time

All algorithms in our scheme is implemented by python program supported by API of Charm, which are performed on a Ubuntu 14.04-amd64 operation system with Intel Core i5 CPU@2.60 GHz, 4 GB RAM and Python 2.73. The simulation of the cloud-fog environment is also defined by software, which could perform the assigned tasks of CSP and fog node. And we also

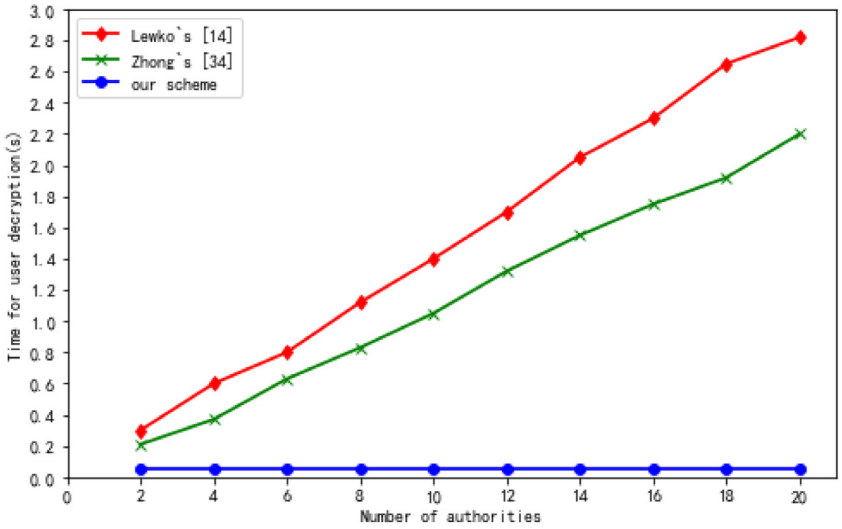
Table 1
Functions comparison.

Scheme	Access structure	Authority	Outsourced decryption	Verifiability	Privacy preserving	Revocation
Phuong's [27]	And Gate	Single	×	×	✓	
Xu's [28]	Tree	Single	×	×	✓	×
Lewko's [14]	LSSS	Multiple	×	×	×	×
Zhong's [34]	LSSS	Multiple	×	×	✓	✓
Our scheme	LSSS	Multiple	✓	✓	✓	✓

×: not supported; ✓: supported.



(a) Outsourced decryption time



(b) User decryption time

Fig. 3. Comparison of decryption time with different number of authority.

simulate the process of communication among the components in our system by python program.

In our experiment, let α_i is an attribute and let $\alpha_1, \alpha_2, \alpha_3, \ldots, \alpha_n$ denote access policies. We have 20 distinct kind of access policy in this form with N increasing from 20 to 200, and repeat each instance 20 times. The results are the average values of every experiment which is repeated 20 times. In addition, the number of attributes per authority is set to 10.

We simulate the decryption computation cost of our scheme, including the cost of outsourced decryption executed by proxy servers in the fog nodes and user decryption executed by end users. Since our scheme is based on Lewko's scheme [14], we compare our scheme with Lewko's scheme and Zhong's scheme [34] in user decryption time.

The time for outsourced decryption are shown in Fig. 3(a) and the time for end user encryption is shown in Fig. 3(b). In Fig. 3(a), the outsourced decryption time is approximately 0.2–2.2 s and it increases almost linearly with the number of the attributes. In Fig. 3(b), the time for end user encryption of our system is almost

constant and much less than that in Lewko's scheme and Zhong's scheme because of outsourced decryption.

In general, we can conclude that our scheme's computation efficiency on the user side is much better than Lewko's scheme and Zhong's scheme. Apparently most computing overheads are outsourced to fog nodes, and only a few computations are left for end users. Therefore, due to the computing cost of end users is greatly reduced, the proposed scheme is highly efficient for fog-enabled IoT system.

7. Conclusion

In this paper, to realize data access control for fog-enabled IoT, we have proposed an efficient and privacy preserving outsourced multi-authority access control scheme, named PPO-MACS. We introduced the one-way anonymous key agreement to make attributes anonymous so as to realizing privacy preserving. And we reduced computation overheads on the end user side by outsourcing part of decryption computation. Meanwhile, we provided an efficient user revocation scheme. Finally, the result of security analysis and performance simulation show that our scheme is secure and highly efficient.

In our future work, we will further discuss the possibility of enhancing the security in the privacy preserving based on ensuring efficiency.

Acknowledgment

This work is supported by the National Key R&D Program of China (No. 2017YFB0802300), the National Natural Science Foundation of China (No. 61772403 and No. U1401251), Natural Science Basic Research Plan in Shaanxi Province of China (No. 2017JM6004), the Fundamental Research Funds for the Central Universities, and National 111 Program of China B16037 and B08038.

References

[1] A. Alshehri, R. Sandhu, Access control models for virtual object communication in cloud-enabled IoT, in: Information Reuse and Integration (IRI), 2017 IEEE International Conference on, IEEE, 2017, pp. 16–25.

[2] A. Alshehri, R. Sandhu, Access Control Models for Cloud-Enabled Internet of Things: A Proposed Architecture and Research Agenda, 2016.

[3] A. Botta, W. De Donato, V. Persico, A. Pescapé, On the integration of cloud computing and internet of things, in: Future Internet of Things and Cloud (FiCloud), 2014 International Conference On, IEEE, 2014, pp. 23–30.

[4] A. Botta, W. De Donato, V. Persico, A. Pescapé, Integration of cloud computing and internet of things: a survey, Future Gener. Comput. Syst. 56 (2016) 684–700.

[5] C. Puliafito, E. Mingozzi, G. Anastasi, Fog computing for the internet of mobile things: issues and challenges, in: Smart Computing (SMARTCOMP), 2017 IEEE International Conference on, IEEE, 2017, pp. 1–6.

[6] F. Bonomi, R. Milito, J. Zhu, S. Addepalli, Fog computing and its role in the internet of things, in: Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing, ACM, 2012, pp. 13–16.

[7] F. Computing, The internet of things: extend the cloud to where the things are, 2015, Available on: http://www.cisco.com/c/dam/en_us/solutions/trends/iot/docs/computingoverview.pdf.

[8] D. Roca, R. Milito, M. Nemirovsky, M. Valero, Tackling IoT Ultra Large Scale Systems: fog computing in support of hierarchical emergent behaviors, in: Fog Computing in the Internet of Things, Springer, 2018, pp. 33–48.

[9] E. Baccarelli, P.G.V. Naranjo, M. Scarpiniti, M. Shojafar, J.H. Abawajy, Fog of everything: Energy-efficient networked computing architectures, research challenges, and a case study, IEEE Access 5 (2017) 9882–9910.

[10] J. Ni, K. Zhang, X. Lin, X. Shen, Securing fog computing for internet of things applications: challenges and solutions, IEEE Commun. Surv. Tutor. (2017).

[11] C. Thota, R. Sundarasekar, G. Manogaran, R. Varatharajan, M. Priyan, Centralized fog computing security platform for IoT and cloud in healthcare system, in: Exploring the Convergence of Big Data and the Internet of Things, IGI Global, 2018, pp. 141–154.

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

[13] M. Chase, Multi-authority attribute based encryption, in: Theory of Cryptography Conference, Springer, 2007, pp. 515–534.

[14] A. Lewko, B. Waters, Decentralizing attribute-based encryption, in: Annual International Conference on the Theory and Applications of Cryptographic Techniques, Springer, 2011, pp. 568–588.

[15] Z. Zhou, D. Huang, Z. Wang, Efficient privacy-preserving ciphertext-policy attribute based-encryption and broadcast encryption, IEEE Trans. Comput. 64 (1) (2015) 126–138.

[16] S. Wang, J. Zhou, J.K. Liu, J. Yu, J. Chen, W. Xie, An efficient file hierarchy attribute-based encryption scheme in cloud computing, IEEE Trans. Inform. Forensics Secur. 11 (6) (2016) 1265–1277.

[17] M. Green, S. Hohenberger, B. Waters, Outsourcing the decryption of abe ciphertexts, in: USENIX Security Symposium, vol. 3, 2011.

[18] K. Yang, X. Jia, Attributed-based access control for multi-authority systems in cloud storage, in: Distributed Computing Systems (ICDCS), 2012 IEEE 32nd International Conference on, IEEE, 2012, pp. 536–545.

[19] K. Yang, X. Jia, Expressive, efficient, and revocable data access control for multi-authority cloud storage, IEEE Trans. Parallel Distrib. Syst. 25 (7) (2014) 1735–1744.

[20] J. Lai, R.H. Deng, C. Guan, J. Weng, Attribute-based encryption with verifiable outsourced decryption, IEEE Trans. Inform. Forensics Secur. 8 (8) (2013) 1343–1354.

[21] X. Mao, J. Lai, Q. Mei, K. Chen, J. Weng, Generic and efficient constructions of attribute-based encryption with verifiable outsourced decryption, IEEE Trans. Dependable Secure Comput. 13 (5) (2016) 533–546.

[22] K. Fan, J. Wang, X. Wang, H. Li, Y. Yang, A secure and verifiable outsourced access control scheme in fog-cloud computing, Sensors 17 (7) (2017) 1695.

[23] C. Zuo, J. Shao, G. Wei, M. Xie, M. Ji, CCA-secure ABE with outsourced decryption for fog computing, Future Gener. Comput. Syst. 78 (2018) 730–738.

[24] Z. Li, W. Li, Z. Jin, Q. Wen, An efficient ABE scheme with verifiable outsourced encryption and decryption, IEEE Access (2019).

[25] T. Nishide, K. Yoneyama, K. Ohta, ABE with Partially Hidden Encryptor-Specified Access Structure. ACNS'08, in: LNCS, vol. 5037, Springer, 2008.

[26] J. Lai, R.H. Deng, Y. Li, Fully secure cipertext-policy hiding CP-ABE, in: International Conference on Information Security Practice and Experience, Springer, 2011, pp. 24–39.

[27] T.V.X. Phuong, G. Yang, W. Susilo, Hidden ciphertext policy attribute-based encryption under standard assumptions, IEEE Trans. Inform. Forensics Secur. 11 (1) (2016) 35–45.

[28] R. Xu, B. Lang, A CP-ABE scheme with hidden policy and its application in cloud computing, Int. J. Cloud Comput. 4 (4) (2015) 279–298.

[29] H. Qian, J. Li, Y. Zhang, J. Han, Privacy-preserving personal health record using multi-authority attribute-based encryption with revocation, Int. J. Inform. Secur. 14 (6) (2015) 487–497.

[30] B. Waters, Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization, in: International Workshop on Public Key Cryptography, 2011, pp. 53–70.

[31] J. Lai, R.H. Deng, Y. Li, Expressive CP-ABE with partially hidden access structures, in: Proceedings of the 7th ACM Symposium on Information, Computer and Communications Security, ACM, 2012, pp. 18–19.

[32] Z. Liu, Z. Cao, Q. Huang, D.S. Wong, T.H. Yuen, Fully secure multi-authority ciphertext-policy attribute-based encryption without random oracles, in: European Symposium on Research in Computer Security, Springer, 2011, pp. 278–297.

[33] 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.

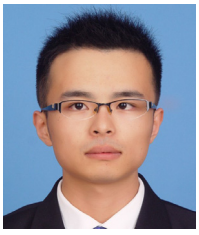
[34] H. Zhong, W. Zhu, Y. Xu, J. Cui, Multi-authority attribute-based encryption access control scheme with policy hidden for cloud storage, Soft Comput. 22 (1) (2018) 243–251.

[35] A. Beimel, Secure Schemes for Secret Sharing and Key Distribution, Technion-Israel Institute of technology, Faculty of computer science, 1996.

[36] A. Kate, G. Zaverucha, I. Goldberg, Pairing-based onion routing, in: International Workshop on Privacy Enhancing Technologies, Springer, 2007, pp. 95–112.

[37] J.A. Akinyele, et al., Charm: a framework for rapidly prototyping cryptosystems, J. Cryptogr. Eng. 3 (2) (2013) 111–128.

Kai Fan received his BS, MS and PhD degrees from Xidian University, P. R. China, in 2002, 2005 and 2007, respectively, in Telecommunication Engineering, Cryptography and Telecommunication and Information System. He is working as an associate professor in State Key Laboratory of Integrated Service Networks at Xidian University. He has published over 40 papers in journals and conferences. He received 3 Chinese patents. He has managed 5 national research projects. His research interests include IoT security, cloud computing security and information security. The mailing address is 2 South Taibai Road, Xidian University, Xian 710071, China. The email address is kfan@mail.xidian.edu.cn.



Huiyue Xu was born in 1992 in Shanxi province of China. He received her B. S. degree in telecommunication engineering from Harbin Institute of Technology in 2016. He is studying as a master in State Key Laboratory of Integrated Service Networks at Xidian University. He is studying privacy preserving in cloud computing and cloud storage.

Longxiang Gao received his PhD in Computer Science from Deakin University, Australia. He is currently a Lecturer at School of Information Technology, Deakin University to teach database, web development and networking units for both undergraduate and postgraduate students. Before joined Deakin University, he was a post-doctoral research fellow at IBM Research & Development Australia. In IBM R&D, he had been the core team member to develop a crisis event analysis, computing and reporting system to Australia Red Cross, and this project has been selected as the feature project of IBM. His research interests include data processing, mobile social networks, fog computing and network security.

Hui Li was born in 1968 in Shaanxi Province of China. In 1990, he received his B. S. degree in radio electronics from Fudan University. In 1993, and 1998, he received his M. S. degree and Ph. D. degree in telecommunications and information system from Xidian University respectively. He is now a professor of Xidian University. His research interests include network and information security.

Yintang Yang was born in 1968 in Shaanxi Province of China. In 1990, he received his B. S. degree in radio electronics from Fudan University. In 1993, and 1998, he received his M. S. degree and Ph. D. degree in telecommunications and information system from Xidian University respectively. He is now a professor of Xidian University. His research interests include network and information security.