IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received June 25, 2017, accepted July 11, 2017, date of publication July 14, 2017, date of current version July 31, 2017.

Digital Object Identifier 10.1109/ACCESS.2017.2727054

Secure Data Access Control With Ciphertext Update and Computation Outsourcing in Fog Computing for Internet of Things

QINLONG HUANG, (Member, IEEE), YIXIAN YANG, AND LICHENG WANG
State Key Laboratory of Networking and Switching Technology, Information Security Center, Beijing University of Posts and Telecommunications, Beijing 100876, China

Corresponding author: Qinlong Huang (longsec@bupt.edu.cn)

This work was supported in part by the National Key Research and Development Program of China under Grant 2016YFB0800605, in part by the National Natural Science Foundation of China under Grant 61572080, and in part by the CCF and Venustech Research Program under Grant 2016012.

ABSTRACT Fog computing is a paradigm that extends cloud computing to the edge of the network. It can provide computation and storage services to end devices in Internet of Things (IoT). Attribute-based cryptography is a well-known technology to guarantee data confidentiality and fine-grained data access control. However, its computational cost in encryption and decryption phase is linear with the complexity of policy. In this paper, we propose a secure and fine-grained data access control scheme with ciphertext update and computation outsourcing in fog computing for IoT. The sensitive data of data owner are first encrypted using attribute-based encryption with multiple policies and then outsourced to cloud storage. Hence, the user whose attributes satisfy the access policy can decrypt the ciphertext. Based on the attribute-based signature technique, authorized user whose attributes integrated in the signature satisfy the update policy can renew the ciphertext. Specifically, most of the encryption, decryption, and signing computations are outsourced from end devices to fog nodes, and thus, the computations for data owners to encrypt, end users to decrypt, re-encrypt, and sign are irrelevant to the number of attributes in the policies. The security analysis shows that the proposed scheme is secure against known attacks, and the experimental results show that the fog nodes perform most of the computation operations of encryption, decryption, and signing, and hence, the time of encryption for data owner, decryption, re-encryption, and signing for users is small and constant.

INDEX TERMS Internet of Things, fog computing, access control, data security, attribute based encryption, attribute based signature.

I. INTRODUCTION

Nowadays, the cloud computing is considered as a promising computing paradigm, since it can provide elastic computing resources to users based on the techniques of distributed computing, virtualization, and so on [1]. However, the prevalence of the Internet of Things (IoT) applications are now changing the main factor of computing [2], [3]. The centralized computing systems are starting to suffer from the unbearable transmission latency and degraded service due to the extraordinary huge volume traffic between IoT devices and cloud. Fog computing is a promising technology that takes advantage of both the paradigms of cloud computing and the IoT, which has the characteristics of location awareness, geo-distribution, low latency, mobility support, etc. [4].

Although the great benefits brought by fog computing paradigm, security problems including data confidentiality and access control are similar to that in the area of cloud computing and IoT. Moreover, they are more easily compromised and low-trustworthy since fog nodes are deployed at the network edge and much lower cost than cloud servers [5]. One promising approach to solving such problems is to encrypt data in advance before the upload. The concept of attribute-based encryption (ABE) is a one-to-many cryptographic technique that fulfills these requirements [6]. It features a mechanism that enables an access control over encrypted data using access policies and ascribed attributes among private keys and ciphertexts. Especially, the ciphertext-policy ABE (CP-ABE) enables data owner to define the access policy over a universe of attributes that the user needs to possess in order to decrypt the ciphertext, and enforce it on the data [7]. In this way, the confidentiality and fine-grained access control of data can be guaranteed.

However, existing ABE-based solutions mainly focus on how to afford secure data access for users, few works consider that there is another requirement that data owner may want to authenticate some users to update the encrypted data [8]. For instance, Alice is a data owner and she outsources the encrypted data to cloud, she hopes that only her several friends who are regarded as valid users can renew the initial ciphertext. Thus, the key point of secure ciphertext update is that the user who renews the ciphertext should be able to prove to the cloud service provider (CSP) that he is a valid user. The traditional approach is to sign the modified data, which means CSP must simultaneously maintain a public key list of valid users to verify the identities of users. However, it would bring a lot of extra burden to maintain the key list if existing a large number of users, and CSP can know the identities of users in this way, which discloses the user privacy. A novel cryptographic technique known as attribute-based signature (ABS) is able to help CSP to verify whether the user is valid [9]–[11]. In an ABS system, user can sign messages with a claim policy and his attributes. Then, with the signature, the CSP can check whether the signer's attributes satisfy the claim policy while remaining completely ignorant of the identity of signer. Therefore, adopting ABE and ABS can achieve data confidentiality, fine-grained access control and user verification, but it also brings high computational cost at the same time in fog computing [12]. The encryption, decryption and signing operations of ABE and ABS require a large number of module exponentiations, which commonly grow linearly with the number of attributes in policies. This presents a significant challenge for users who access and modify data on resource-constrained IoT devices with limited computation and storage capacity.

In this paper, we propose a secure data access control scheme in fog computing for IoT. The main contributions are as follows:

1) We propose a fine-grained data access control scheme with ciphertext update based on CP-ABE and ABS in fog computing. First, the sensitive data from IoT devices are encrypted with multiple policies and then outsourced to cloud servers through nearby fog nodes. The authorized user whose attributes satisfy the access policy can decrypt the ciphertext stored in the cloud servers. Second, the authorized user can modify the decrypted data and re-outsource it again with his signature. If the user's attributes in the signature satisfy the update policy, the cloud servers can renew the ciphertext.

2) We provide a secure outsourcing construction which outsources most of encryption, decryption and signing computations from end IoT devices to fog nodes, thus the computations for data owners to encrypt, end users to decrypt, re-encrypt and sign are irrelevant to the number of attributes in the policies.

The experimental results show that fog nodes perform the heavy computation operations of encryption, decryption and signing, hence the time of encryption for data owner, decryption, re-encryption and signing for users is small and constant. This paper is structured as follows. We review related work in Section II, introduce the preliminaries and definitions in Section III, and provide the system model, system definition and security model in Section IV. The detailed construction of algorithms is given in Section V, and the security and performance of our scheme are analyzed in Section VI and VII respectively. Finally, we conclude this paper in Section VIII.

II. RELATED WORKS

The concept of fog computing is proposed by Cisco in 2014, which can be regarded as a layer in the middle of the cloud and end users consisted by fog nodes, such as hardened routers, switches, and etc. [13]. They are much closer to end users than cloud servers, and some of the workloads and services taken in the cloud are moved to the fog nodes. Similar to the cloud servers, fog nodes are not fully trusted as well, data security would raise great concerns from users when they store sensitive data on cloud servers through fog nodes [14]. Thus, a new access control scheme with cloud, fog and users should be considered, since the network structures and system models are different, in which fog nodes should assist user, to make less computational complexity and more flexibility left for users.

ABE is a promising cryptographic technique to realize scalable, flexible, and fine-grained access control solutions. The notion of ABE was first introduced by Sahai and Waters as a new method for fuzzy identity-based encryption [15], [16]. ABE has two variants, key-policy ABE (KP-ABE) [17] and CP-ABE [18]. Actually, it becomes a powerful mechanism that can be applied to realize access control in many applications in IoT [19]–[22]. Yu et al. [20] introduced the fine-grained data access control problem in wireless sensor networks for the first time, and they adopted KP-ABE to protect data. In contrast to KP-ABE, CP-ABE turns out to be well suited for access control in IoT due to its expressiveness in describing access policy of ciphertext. Hu et al. [21] designed a secure data communication scheme between wearable sensors and data consumers by employing CP-ABE in wireless body area networks. Jiang et al. introduced a CP-ABE scheme against key-delegation abuse in fog computing [22]. Yeh et al. [23] proposed a fine-grained health information access control framework in the cloud for lightweight IoT devices.

However, the most significant drawback of ABE for the use in fog computing is the computational cost in the encryption and decryption phase which is linear with the complexity of policy. Fog nodes, the edge of the cloud and closer to end users, are one of the best choices for the outsourcing proxy [24], [25], which can be used to do massive computations to reduce the computational overhead required on resource-constrained IoT devices. The main solution of current schemes is to distribute calculations of CP-ABE encryption and decryption phase, so that constrained IoT devices can delegate most of the consuming operations to nodes of the network [26]–[31]. Lounis et al. [28] designed a cloud-based architecture for medical WSNs, in which sensor nodes outsource the encryption operations to a trusted gateway that encrypts data based on CP-ABE before sending to cloud. However, this solution adopts a full trusted entity to perform data encryption which does not achieve practical computation outsourcing. Zuo et al. [29] designed a concrete ABE scheme with outsourced decryption for fog computing. Yang et al. [30] proposed a concrete construction with lightweight computational overhead for health IoT system, in which a semi-trusted computation center is introduced to enforce most of the heavy calculations in data encryption phase. Yang et al. [31] proposed two multiple cloud based ABE schemes for IoT, which enable receivers to partially outsource computationally expensive decryption operations to the clouds. However, these schemes only can support either the outsourced encryption or outsourced decryption. Zhang et al. proposed an access control scheme for fog computing, which outsources the heavy computation of encryption and decryption to fog nodes, thus the computations for data owners to encrypt and users to decrypt are irrelevant to the number of attributes in access policy [32].

In order to realize ciphertext update services in fog computing, the CSP must have the ability to verify the user's proof before accepting the modified ciphertext. ABS is an emerging signature algorithm to ensure anonymous user authentication. It was first introduced by Maji et al. [33], in order to provide authentication without disclosing the identities of the users. Based on ABS, Ruj et al. proposed a new decentralized access control scheme for secure data read and write in clouds, which supports anonymous user authentication [11]. In this scheme, the cloud verifies the authenticity without knowing the user's identity before storing data. Su et al. [34] proposed an expressive ABS scheme in IoT, which uses an attribute tree to assurance that only a user with appropriate attributes satisfying the access policy can endorse the message. However, in the existing works of ABS, heavy computational cost is required during the signing phase, which also grows linearly with the size of predicate formula. Chen et al. are the first to present two outsourced ABS schemes in which the computational overhead at user side is greatly reduced through outsourcing intensive computations to untrusted CSP [35]. Inspired by this, our scheme realizes anonymous user authentication in ciphertext update phase and delegates most of the signing operations to the fog nodes.

III. PRELIMINARIES AND DEFINITIONS

A. BILINEAR MAP

Let $\mathbb{G}_0$ and $\mathbb{G}_T$ be two multiplicative groups of prime order p. A bilinear map is a function $e : \mathbb{G}_0 \times \mathbb{G}_0 \rightarrow \mathbb{G}_T$ with the following properties:

1) Computability. There is an efficient algorithm to compute $e(g, h) \in \mathbb{G}_T$, for any $g, h \in \mathbb{G}_0$.
2) Bilinearity. For all $g, h \in \mathbb{G}_0$ and $a, b \in \mathbb{Z}_p$, we have $e(g^a, h^b) = e(g, h)^{ab}$.
3) Non-degeneracy. If g is a generator of $\mathbb{G}_0$, then $e(g, g)$ is also a generator of $\mathbb{G}_T$.

B. ACCESS TREE

Let T denote a tree, a logical representation of an access policy. Each non-leaf node x represents a threshold gate, described by its children and a threshold value. Let num_x denote the number of children of a node x, and k_x represent its threshold value. For each leaf node y, we have $k_y = 1$. Let $attr_y$ denote an attribute associated with leaf node y in the tree and $parent(z)$ represent a parent node of the node z in the tree. Each child node of the node x in the tree is labelled from 1 to num_x, and $index(x)$ returns such label associated with the node x. These index values are uniquely assigned to nodes in the access tree for a given key in an arbitrary manner.

Let T_x be a subtree of T rooted at the node x. If a set of attributes r satisfies the access tree T_x, we denote it as $T_x(r) = 1$. We compute $T_x(r)$ recursively as follows. For a non-leaf node x, it evaluates $T_{x'}(r)$ for all children x' of node x. For non-leaf node x, $T_x(r)$ returns 1 if and only if at least k_x children return 1. For the leaf node y, it returns 1 if and only if $attr_y \in r$.

C. CIPHERTEXT-POLICY ATTRIBUTE-BASED ENCRYPTION

A CP-ABE system for access policy T consists of the following four algorithms.

1) *Setup*(1^κ)*:* The setup algorithm takes as input the security parameter κ and outputs a public key PK and a master secret key MK.
2) *KeyGen(PK, MK, S):* The key generation algorithm takes as input the public key PK, the master secret key MK, a set S of attributes, and outputs a secret key SK.
3) *Enc(PK, M, T):* The encryption algorithm takes as input the public key PK, a message M and an access policy T, and outputs a ciphertext CT.
4) *Dec(PK, SK, CT):* The decryption algorithm takes as input the public key PK, a secret key SK, a ciphertext CT with an access policy T. If $S \in T$, it outputs the message M.

D. ATTRIBUTE-BASED SIGNATURE

The ABS scheme consists of four algorithms as follows:

1) *Setup*(1^κ)*:* The system setup is the algorithm run by the attribute authority for which the input is the security parameter κ and the outputs are public key PK and master secret key MK.
2) *KeyGen(PK, MK, S):* The key generation is the algorithm run by the attribute authority on inputs public key PK, master secret key MK and a set of attributes S to generate the secret key SK for the signer.
3) *Sign(PK, M, T, SK):* The signing is the algorithm run by a signer on inputs PK, a message M, a claim policy T and secret key SK to generate a signature ST for the message.

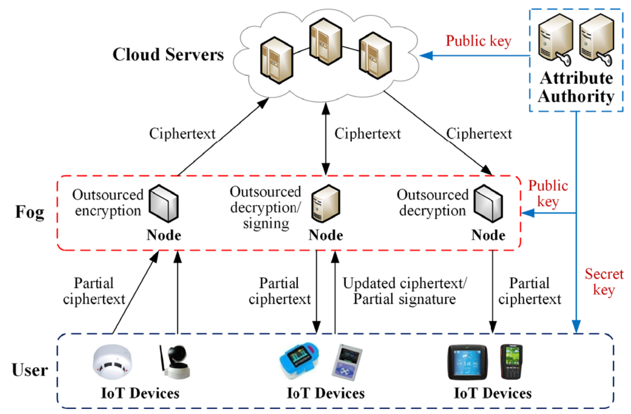


FIGURE 1. System model.

4) *Verify(PK, M, T, ST):* The verifying is the algorithm run by a verifier on inputs *PK*, a message M, a claim policy T and a signature *ST*. The output is true if *ST* is a valid signature by a signer whose attributes satisfying T.

IV. SYSTEM MODEL AND SECURITY MODEL

A. SYSTEM MODEL

The system model of our proposed scheme consists of attribute authority, CSP, fog nodes, data owners and users, as shown in Fig. 1.

1) Attribute authority. The attribute authority is a fully trusted party which is in charge of generating system parameters as well as secret key for each user.
2) CSP. The CSP is a semi-trusted party which provides high-capacity and online data storage service. It is also responsible for verifying the signature before accepting the updated ciphertext.
3) Fog node. The fog nodes are also semi-trusted parties which are deployed at the network edge and offer a variety of services. They are in charge of generating part of the ciphertext and uploading the whole ciphertext to the CSP, and also helping users to decrypt the ciphertext from the CSP. Moreover, they assist end users to sign the ciphertext update request.
4) Data owner. The data owner has a great amount of data from the IoT devices to be uploaded to cloud. It is designed to define access and update policies to generate the whole ciphertext with the fog nodes.
5) User. The user is attached to fog nodes and equipped with IoT devices such as smart cameras, medical sensors and smart meters [36]. Since the IoT device has limited computation and storage ability, it wishes to gain access to the ciphertext stored in CSP with the help of fog nodes. If the user's attribute set satisfies the access policy in the ciphertext, he is able to decrypt the underlying data. After accessing the data, the user may make a modification and wish to re-encrypt the data. If the user's attribute set satisfies the update policy in the ciphertext, the CSP will renew the stored ciphertext.

B. SYSTEM DEFINITION

We define our proposed scheme by describing the following five phases and nine algorithms.

Phase 1 (System Setup):

1) *Setup*(1^κ)*:* The attribute authority takes as input security parameter κ, and outputs the system public key *PK* and master secret key *MK*.

Phase 2 (Key Generation):

2) *KeyGen*(PK, MK, S)*:* The attribute authority takes as input *PK*, *MK*, a set of attributes S, outputs the secret key *SK* for the user. And the outsourcing key SK' is sent to fog nodes.

Phase 3 (Data Encryption):

3) *Fog.Encrypt*(PK, T_a)*:* The fog node takes as input *PK*, an access policy T_a, outputs a partial ciphertext CT'.

4) *Owner.Encrypt*(PK, M, T_u, CT')*:* The data owner takes as input *PK*, a data M, an update policy T_u, a partial ciphertext CT', and outputs the ciphertext *CT*.

Phase 4 (Data Decryption):

5) *Fog.Decrypt*(PK, CT, SK')*:* The fog node takes as input *PK*, a ciphertext *CT* and a user's SK', and outputs a partial decrypted ciphertext T if the attributes satisfy access policy T_a in the ciphertext *CT*.

6) *User.Decrypt*(T, SK)*:* The user takes as input a partial decrypted ciphertext T and *SK*, then recovers the *DK* and outputs the plaintext M.

Phase 5 (Ciphertext Update):

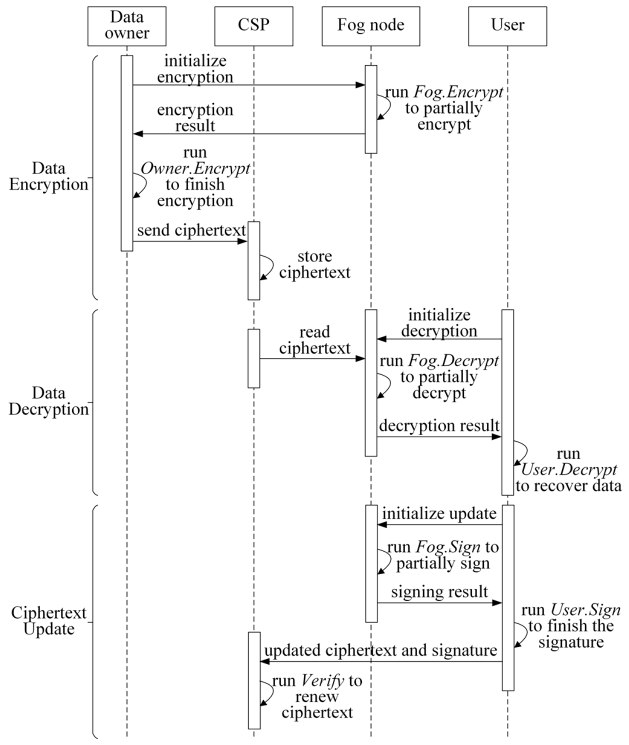
7) *Fog.Sign*(PK, U, T_u, SK')*:* The fog node takes as input *PK*, a user's ciphertext update request U and SK', update policy T_u. It outputs a partial signature ST' and the global key *GK*.

8) *User.Sign*(PK, ST', SK)*:* The user takes as input *PK*, a partial signature ST' and *SK*, outputs the signature *ST*.

9) *Verify*(PK, ST, GK)*:* The CSP takes as input *PK*, a signature *ST* and a global key *GK*. It outputs true if *ST* is a valid signature by the signer whose attributes satisfying T_u.

The work flow of our scheme is shown in Fig. 2. At the initialization phase, attribute authority uses the *Setup* algorithm to generate systems parameter. By the *KeyGen* algorithm, attribute authority generates secret keys for data owners and users. In order to achieve high encryption efficiency, the data owner first encrypts the collected data with a random *DK* by applying symmetric encryption algorithm and defines an access policy and an update policy, the fog node uses the *Fog.Encrypt* algorithm to partially encrypt the data with the access policy, and then data owner uses the *Owner.Encrypt* algorithm to finish the encryption with both the access policy and update policy and stores it to the CSP. When accessing the data, the fog node first uses the *Fog.Decrypt* algorithm to partially decrypt the ciphertext, and then the user can use the *User.Decrypt* algorithm to recover the data.

After modifying the data, user also uses the algorithms in the encryption phase to encrypt the updated data. Before making the final modification, user uses the *User.Sign* algorithm to generate the signature with the partial signature returned from fog node which runs the *Fog.Sign* algorithm. Then the



Data owner
CSP
Fog node
User
initialize encryption
run *Fog.Encrypt* to partially encrypt
encryption result
run *Owner.Encrypt* to finish encryption
send ciphertext
store ciphertext
Data Encryption
initialize decryption
read ciphertext
run *Fog.Decrypt* to partially decrypt
decryption result
run *User.Decrypt* to recover data
Data Decryption
initialize update
run *Fog.Sign* to partially sign
signing result
run *User.Sign* to finish the signature
updated ciphertext and signature
run *Verify* to renew ciphertext
Ciphertext Update

FIGURE 2. Work flow of our scheme.

CSP uses the *Verify* algorithm to verify the signature and finally accepts the updated ciphertext if the signature is true. In the end, other users can obtain the updated data with the decryption algorithms. Therefore, the users with IoT devices can access and update confidential data in fog computing efficiently.

C. SECURITY MODEL

In our scheme, we assume that cloud servers and fog nodes are honest but curious, which means they execute the tasks and may collude to get the unauthorized data. Specifically, the security model covers the following aspects.

1) Data confidentiality. The unauthorized users which are not the intended receivers defined by data owner should be prevented from accessing the data.
2) Fine-grained access control. The data owner can custom expressive and flexible policies so that the data only can be accessed and updated by the users whose attributes satisfy these policies.
3) Authentication. If users could not satisfy the update policy in ciphertexts, it should also be prevented from updating the ciphertexts.
4) Collusion resistance. Two or more users cannot combine their secret and outsourcing keys and get access to the data they cannot access individually.

V. CONSTRUCTION OF ALGORITHMS

In fog computing, it is the essential requirement to make less computational complexity, since most of the IoT devices are resource-constrained. First, we propose fine-grained data access control and efficient ciphertext update scheme based on CP-ABE and ABS. The authorized users whose attributes satisfy the access policy can decrypt the ciphertext, and satisfy the update policy can renew the ciphertext. Second, we provide a secure outsourcing construction which outsources most of encryption, decryption and signing computations from end IoT devices to fog nodes. The construction details are as follows.

A. SYSTEM SETUP

The attribute authority runs *Setup* algorithm to select a bilinear map $e : \mathbb{G}_0 \times \mathbb{G}_0 \rightarrow \mathbb{G}_T$, where $\mathbb{G}_0$ and $\mathbb{G}_T$ are two multiplicative groups with prime order p, and g is the generator of $\mathbb{G}_0$. Then the attribute authority randomly chooses $h \in \mathbb{G}_0$ and $\alpha, \beta \in \mathbb{Z}_p$, chooses cryptographic hash functions $H_1 : \{0, 1\}^* \rightarrow \mathbb{Z}_p^*$, $H_2 : \{0, 1\}* \rightarrow \mathbb{G}_0$, finally outputs a system public key $PK = (g, h, g^\alpha, g^\beta, h^\beta, e(g, g)^{\alpha\beta})$ and a master secret key $MK = (\alpha, \beta)$.

B. KEY GENERATION

The attribute authority runs *KeyGen* algorithm to select a random $\gamma \in \mathbb{Z}_p$, which is a unique secret assigned to each user. Then the attribute authority chooses a random $\varepsilon \in \mathbb{Z}_p$, and random r_j for each attribute $j \in S$, where S is the attribute set of user, and outputs the secret key and outsourcing key.

$$\begin{aligned} SK &= (D = g^{(\alpha+\gamma)\beta}) \\ SK' &= (D_1 = g^\gamma h^\varepsilon, D_2 = g^\varepsilon, \\ &\qquad \{\tilde{D}_j = g^{\gamma\beta} H_1(j)^{r_j}, \tilde{D'}_j = g^{r_j}\}_{j\in S}) \end{aligned} \tag{1}$$

The outsourcing key $SK' = (D_1, D_2, \{\tilde{D}_j, \tilde{D'}_j\}_{j\in S})$ of user is sent to the fog nodes, and the user only stores SK.

C. DATA ENCRYPTION

Before uploading data to the CSP, data owner first chooses a random $DK \in \mathbb{Z}_p$, and encrypts the data M with DK using symmetric encryption algorithm, denoted as $C = SE_{DK}(M)$. Then data owner defines an access policy T_a and an update policy T_u, and sends T_a to fog nodes.

The fog nodes run *Fog.Encrypt* algorithm to perform the outsourced encryption. For each node x in access policy tree T_a, the fog nodes choose a polynomial p_x. Beginning from the root node R, the p_x is chosen in a top-down manner. For each node x in the tree, set the degree d_x of the polynomial p_x to be one less than the threshold value k_x of that node, that is $d_x = k_x - 1$.

Starting with the root node R, the algorithm chooses a random $s \in \mathbb{Z}_p$ and sets $p_R(0) = s$. Then, it chooses d_R other points of the polynomial p_R randomly to define it completely. For any other node x, it sets $p_x(0) = p_{parent(x)}(index(x))$ and chooses d_x other points randomly to completely define p_x. Let Y be the set of leaf nodes in T_a, the fog nodes output a partial ciphertext CT'.

$$\begin{aligned} CT' &= (T_a, C'_3 = g^{\beta s}, C'_4 = h^{\beta s}, \\ C_5 &= \{\tilde{C}_y = g^{p_y(0)}, \tilde{C'}_y = H_1(attr_y)^{p_y(0)}\}_{y\in Y}) \end{aligned} \tag{2}$$

Finally, the fog nodes return CT' to the data owner. The data owner runs *Owner.Encrypt* algorithm to select $t \in \mathbb{Z}_p$ at random and computes $C_1 = DK \cdot e(g,g)^{\alpha\beta t}$ with DK, and computes $C_2 = g^t$, $C_3 = C_3' \cdot g^{\beta t}$, $C_4 = C_4' \cdot h^{\beta t}$. Finally, the data owner outputs the ciphertext CT.

$$\begin{aligned} CT &= (T_a, T_u, C = SE_{DK}(M), C_1 = DK \cdot e(g,g)^{\alpha\beta t},\\ &\quad C_2 = g^t, C_3 = g^{\beta(s+t)}, C_4 = h^{\beta(s+t)},\\ &\quad C_5 = \{\tilde{C}_y = g^{p_y(0)}, \tilde{C'}_y = H_1(attr_y)^{p_y(0)}\}_{y\in Y}) \end{aligned} \tag{3}$$

D. DATA DECRYPTION

If attributes of the user satisfy the access policy T_a, he can decrypt CT successfully by running the following decryption algorithm and obtain the symmetric key DK. Fog nodes run *Fog.Decrypt* algorithm to obtain ciphertext from the CSP. The fog nodes first run *DecryptNode* algorithm which is a recursive algorithm. The algorithm takes a ciphertext CT, SK', and a node x from the access tree T_a as input.

1) If the node x is a leaf node, then we let $z = attr_x$ and define as follows. If $z \in S$, then

$$\begin{aligned} DecryptNode(CT, SK', x) &= \frac{e(\tilde{D}_z, \tilde{C}_x)}{e(\tilde{D'}_z, \tilde{C'}_x)}\\ &= \frac{e(g^{\gamma\beta} H_1(z)^{r_z}, g^{p_x(0)},)}{e(g^{r_z}, H_1(attr_x)^{p_x(0)})}\\ &= e(g,g)^{\gamma\beta p_x(0)} \end{aligned} \tag{4}$$

If $z \notin S$, then $DecryptNode(CT, SK', x) = \perp$.

2) If the node x is a non-leaf node, the algorithm $DecryptNode(CT, SK', x)$ proceeds as follows: for all nodes n that are children of x, it calls $DecryptNode(CT, SK', n)$ and stores the output as F_n. Let S_x be an arbitrary k_x-sized set of child nodes n such that $F_n \neq \perp$. If no such set exists, then the node is not satisfied and the function returns $\perp$. Otherwise, computes and returns the result.

$$\begin{aligned} F_x &= \prod_{n\in S_x} F_n^{\Delta_{j,S_x'}(0)}\\ &= \prod_{n\in S_x} (e(g,g)^{r\beta \cdot p_{parent(n)}(index(n))})^{\Delta_{j,S_x'}(0)}\\ &= \prod_{n\in S_x} e(g,g)^{r\beta \cdot p_x(j) \cdot ^{\Delta_{j,S_x'}(0)}}\\ &= e(g,g)^{r\beta \cdot p_x(0)} \end{aligned} \tag{5}$$

Let $j = index(n)$ and $S_x' = \{index(n) : n \in S_x\}$. If the access policy tree T_a is satisfied by S, we set the result of entire evaluation for the access tree T_a as F.

$$\begin{aligned} F &= DecryptNode(CT, SK', R) = e(g,g)^{\gamma\beta p_R(0)}\\ &= e(g,g)^{\gamma\beta s} \end{aligned} \tag{6}$$

Then, the fog nodes compute

$$B = \frac{e(D_1, C_3)}{e(D_2, C_4)} = \frac{e(g^{\gamma} h^{\varepsilon}, g^{\beta(s+t)})}{e(g^{\varepsilon}, h^{\beta(s+t)})} = e(g,g)^{\gamma\beta(s+t)} \tag{7}$$

and

$$A = B/F = e(g,g)^{\gamma\beta(s+t)}/e(g,g)^{\gamma\beta s} = e(g,g)^{\gamma\beta t} \tag{8}$$

Finally, the fog nodes send a partial ciphertext $T = (T_a, T_u, C = SE_{DK}(M), C_1 = DK \cdot e(g,g)^{\alpha\beta t}, C_2 = g^t, A = e(g,g)^{\gamma\beta t})$ to the user. After receiving T from fog nodes, the user runs the *User.Decrypt* algorithm to obtain the symmetric key DK.

$$DK = \frac{C_1 \cdot A}{e(C_2, D)} = \frac{DK \cdot e(g,g)^{\alpha\beta t} \cdot e(g,g)^{\gamma\beta t}}{e(g^t, g^{(\alpha+\gamma)\beta})} \tag{9}$$

Thus, SE_{DK} (M) can be decrypted with DK by applying the symmetric decryption algorithm.

E. CIPHERTEXT UPDATE

After modifying the decrypted data, the user re-encrypts the modified data as described in data encryption phase, then signs the ciphertext update request with his attributes. Only if the user's attributes in the signature satisfy the update policy T_u, the ciphertext will be authorized to be renewed by CSP.

The user sends the request U, and update policy T_u to fog nodes. The fog nodes run *Fog.Sign* algorithm to perform the outsourced signing. For each node x in the update policy tree T_u, the fog nodes choose a polynomial q_x. Beginning from the root node R, the q_x is chosen in a top-down manner. For each node x in the tree, set the degree d_x' of the polynomial q_x to be one less than the threshold value k_x' of that node, that is $d_x' = k_x' - 1$. Starting with the root node R, the algorithm chooses a random $r \in \mathbb{Z}_p$ and sets $q_R(0) = r$. Then, it chooses d_R' other points of the polynomial q_R randomly to define it completely. For any other node x, it sets $q_x(0) = q_{parent(x)}(index(x))$ and chooses d_x' other points randomly to completely define q_x. Let Z be the set of leaf nodes in T_u, the fog nodes output a global key GK.

$$GK = \{\tilde{K}_z = g^{q_z(0)}, \tilde{K'}_z = H_1(attr_z)^{q_z(0)}\}_{z\in Z} \tag{10}$$

For each $j \in Z$, the fog nodes choose $t_j \in \mathbb{Z}_p$ randomly and compute the following with SK'.

1) If $j \in S \cap Z$, then computes

$$\begin{aligned} \tilde{S}_j &= (\tilde{D}_j \cdot H_1(j)^{t_j})^{1/r} = g^{\gamma\beta/r} H_1(j)^{(r_j+t_j)/r},\\ \tilde{S'}_j &= (\tilde{D'}_j \cdot g^{t_j})^{1/r} = g^{(r_j+t_j)/r} \end{aligned} \tag{11}$$

2) If $j \in Z/S \cap Z$, then computes

$$\tilde{S}_j = (H_1(j)^{t_j})^{1/r} = H_1(j)^{t_j/r}, \tilde{S'}_j = (g^{t_j})^{1/r} = g^{t_j/r} \tag{12}$$

Then, the fog nodes select $\lambda \in \mathbb{Z}_p$ at random and output the partial signature ST'.

$$ST' = (U, S_1' = H_2(U)^{\lambda}, S_2' = g^{\lambda}, S_3 = \{\tilde{S}_j, \tilde{S'}_j\}_{j\in Z}) \tag{13}$$

Finally, the fog nodes return ST' to the user. Then the user runs *User.Sign* algorithm to select $\mu \in \mathbb{Z}_p$ at random and compute $S_1 = S_1' \cdot H_2(U)^{\mu} \cdot D$, $S_2 = S_2' \cdot g^{\mu}$. Finally, the user outputs the signature ST.

$$ST = (U, S_1 = H_2(U)^{\lambda+\mu} \cdot g^{(\alpha+\gamma)\beta}, S_2 = g^{\lambda+\mu}, S_3) \quad (14)$$

If attributes of the user satisfy the update policy T_u stored in the initial ciphertext, the CSP can verify the signature by running the *Verify* algorithm. The CSP first runs *VerifyNode* algorithm which can be described as a recursive algorithm. The algorithm takes a signature ST, GK and a node x from the update tree T_u as input.

1) If the node x is a leaf node, then we let $z = attr_x$ and define as follows. If $z \in S \cap Z$, then

$$\begin{aligned} &VerifyNode(ST, GK, x) \\ &= \frac{e(\tilde{S}_z, \tilde{K}_x)}{e(\tilde{S'}_z, \tilde{K'}_x)} \\ &= \frac{e(g^{\gamma\beta/r} H_1(z)^{(r_z+t_z)/r}, g^{q_x(0)},)}{e(g^{(r_z+t_z)/r}, H_1(attr_x)^{q_x(0)})} \\ &= e(g,g)^{\gamma\beta/r \cdot q_x(0)} \end{aligned} \quad (15)$$

If $z \in Z/S \cap Z$, then

$$\begin{aligned} &VerifyNode(ST, GK, x) \\ &= \frac{e(\tilde{S}_z, \tilde{K}_x)}{e(\tilde{S'}_z, \tilde{K'}_x)} \\ &= \frac{e(H_1(z)^{t_z/r}, g^{q_x(0)},)}{e(g^{t_z/r}, H_1(attr_x)^{q_x(0)})} \\ &= 1 \end{aligned} \quad (16)$$

2) If the node x is a non-leaf node, the algorithm $VerifyNode(ST, GK, x)$ proceeds as follows: for all nodes n that are children of x, it calls $VerifyNode(ST, GK, x)$ and stores the output as I_n. We also let S_x be an arbitrary k_x-sized set of child nodes n such that $I_n \neq \perp$. If no such set exists, then the node is not satisfied and the function returns $\perp$. Otherwise, it computes and returns the result.

$$\begin{aligned} I_x &= \prod_{n \in S_x} I_n^{\Delta_{j,S_x'}(0)} \\ &= \prod_{n \in S_x} (e(g,g)^{\gamma\beta/r \cdot q_{parent(n)}(index(n))})^{\Delta_{j,S_x'}(0)} \\ &= \prod_{n \in S_x} e(g,g)^{\gamma\beta/r \cdot q_x(j) \cdot \Delta_{j,S_x'}(0)} \\ &= e(g,g)^{\gamma\beta/r \cdot q_x(0)} \end{aligned} \quad (17)$$

If the update policy tree T_u is satisfied by S, we set the result of entire evaluation for the update tree T_u as I.

$$\begin{aligned} I &= VerifyNode(ST, GK, R) = e(g,g)^{\gamma\beta/r \cdot q_R(0)} \\ &= e(g,g)^{\gamma\beta} \end{aligned} \quad (18)$$

Then, the CSP checks the following equation.

$$\begin{aligned} \frac{e(g, S_1)}{e(H_2(U), S_2) \cdot I} &= \frac{e(g, H_2(U)^{\lambda+\mu} \cdot g^{(\alpha+\gamma)\beta})}{e(H_2(U), g^{\lambda+\mu}) \cdot e(g,g)^{\gamma\beta}} \\ &= e(g,g)^{\alpha\beta} \end{aligned} \quad (19)$$

If this equation holds, then the CSP accepts the signature, which indicates the CSP will accept the updated ciphertext from this user whose attributes satisfy the update policy. Otherwise, the CSP rejects the user's ciphertext update request.

VI. SECURITY ANALYSIS

If there exists a probabilistic polynomial time (PPT) adversary can win our scheme with non-negligible advantage, then there is a PPT algorithm that can distinguish a decisional bilinear Diffie-Hellman (DBDH) tuple from a random tuple, as proofed in [32]. Hence, our scheme is secure to the DBDH assumption. We analyze the security properties of our scheme as follows.

A. DATA CONFIDENTIALITY

The data is first encrypted using the access policy and update policy, and the confidentiality of the data can be guaranteed against users which don't hold a set of attributes that satisfy the access policy. In encryption phase, though the fog node performs encryption computations for user, it still cannot access the data without the secret key. During the decryption phase, since the set of attributes cannot satisfy the access policy in the ciphertext, the cloud servers or fog nodes cannot recover the value $A = e(g,g)^{\gamma\beta t}$ to further get desired value DK, because it does not know the D of user. Therefore, only the users with valid attributes that satisfy the access policy can decrypt the ciphertext.

B. FINE-GRAINED ACCESS CONTROL

Fine-grained access control allows flexibility in specifying differential access rights of individual users. To enforce this kind of access control, we utilize CP-ABE to escort the symmetric encryption key. In the encryption phase of our scheme, the data owner is able to enforce an expressive and flexible access policy and encrypt the symmetric key which is used to encrypt the data, then outsource the ciphertext to cloud servers. Specifically, the access policy of encrypted data defined in access tree supports complex operations including both AND and OR gate, which is able to represent any desired attribute set. Thus, such construction achieves fine-grained access control.

C. AUTHENTICATION

Our scheme exploits ABS to achieve ciphertext update with authentication, even adversary may try to forge a signature with the update policy that his attributes do not satisfy. Let F be an adversary who makes at most q_{H_1}, q_{H_2}, q_O, q_K and q_S queries to random oracles H_1, H_2, outsourcing key generation oracle, secret keys generation oracle and signing oracle respectively, and produces a successful forgery against our scheme with a non-negligible probability ρ. Then there exists an algorithm B that solves the computational Diffie-Hellman (CDH) problem with a non-negligible probability $\rho' = \rho / q_{H_2}$ [12].

D. COLLUSION RESISTANCE

The users may intend to combine their secret keys and outsourcing keys to access the data which they cannot access individually. In our scheme, attribute authority generates secret keys for different users, the secret key is associated with random γ, which are uniquely related to each user and make the combination of components in different secret keys meaningless. Suppose two or more users with different attributes combine together to satisfy the access policy, they cannot compute $F = e(g, g)^{\gamma\beta s}$ in the outsourced decryption phase. Thus, the proposed scheme is collusion-resistant.

VII. PERFORMANCE ANALYSIS

We analyze the efficiency of our proposed scheme in this section. We will focus on the performance efficiency and implement experiments to evaluate the performance.

TABLE 1. Computational complexity on device.

Schemes	Data encryption	Data decryption	Ciphertext update
[34]	-	-	$(2N_C+2)T_0$
[11]	$3N_CT_0+(2N_C+1)T_E$	$2N_CT_P$	$(2N_C+2)T_0+2N_CT_E$
[29]	$(2N_C+1)T_0$	T_E	-
[32]	$3T_0+T_E$	T_P	-
Our Scheme	$2T_0+2T_E$	T_P	$2T_0$

A. PERFORMANCE EFFICIENCY

Here we analyze the performance efficiency of our scheme with the several IoT-based and fog-based data sharing schemes based on ABS or ABE, in terms of computational complexity on user when performing encryption, decryption and signing. The comparison result is showed in Table 1. Let T_P be the computational cost of a single pairing, T_0 be the computational cost of an exponent operation in $\mathbb{G}_0$,$T_E$be the time for an exponent operation in $\mathbb{G}_T$, N_C be the number of attributes in a ciphertext. We ignore the simple multiplication, hash, symmetric encryption and decryption operations.

In the data encryption phase, since Ruj et al. [11] and Zuo et al. [29] perform full ABE algorithm on local, their encryption computational cost of data owner are $3N_CT_0 + (2N_C + 1)T_E$ and $(2N_C + 1)T_0$ respectively which both grow linearly with the number of attributes in access policy. In our scheme, the constrained IoT device only needs to cost constant time to encrypt the data with the help of fog nodes, which is similar with Zhang et al. [32]. However, Zhang et al. [32] cannot support ciphertext update. The similar situation appears in data decryption phase. From the end user's point of view, computational time in our scheme and

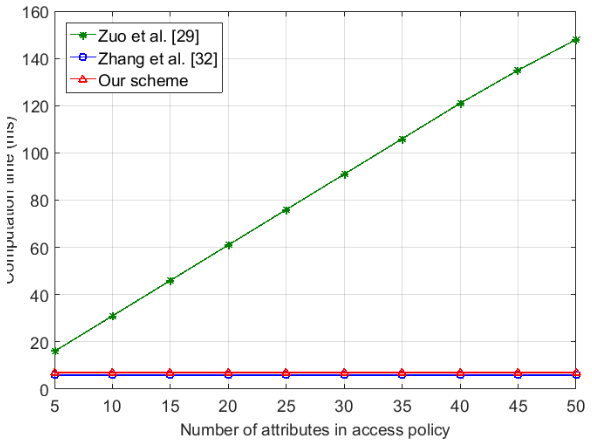


FIGURE 3. Comparison of computational overhead for encryption.

Zhang et al. [32] is lower than that of Ruj et al. [11] since the user only needs one pairing operation to recover the plaintext. Further, in the ciphertext update phase, compared with Ruj et al. [11] and Su et al. [34] which adopt standard ABS to support the update of outsourced ciphertext. The users in our scheme only need to perform two exponentiation operation in $\mathbb{G}_0$ to sign the ciphertext before sending to fog nodes. Thus, the signing cost in our scheme is less than that of Ruj et al. [11] and Su et al. [34] which cost $(2N_C + 2)T_0 + 2N_CT_E$ and $(2N_C + 2)T_0$ respectively.

B. EXPERIMENTAL ANALYSIS

We conduct simulation experiments on a laptop as fog node and an android phone as IoT device. The laptop is with Intel CPU at 2.53 GHz, 4 GB memory and Ubuntu 16.04. The android phone is Samsung G9600V with a quad core processor, 2 GB memory, and Android 6.0.1. The experimental code uses the pairing-based cryptography library [37] to simulate the schemes. We use a pairing-friendly type-A 160-bit elliptic curve group based on the supersingular curve $y^2 = x^3 + x$ over a 512-bit finite field. The Advanced Encryption Standard (AES) is chosen as the symmetric key encryption scheme.

The number of attributes used in the experiments is from 5 to 50, and the experimental result is the average number of 10 runs. We consider this range to be representative enough for a wide range of real world IoT applications.

First, we analyze the time cost of the data encryption and decryption by comparing our scheme with Zuo et al. [29] and Zhang et al. [32]. In data encryption phase, a data owner encrypts a file with an access policy and an update policy, and posts the encrypted file to the public cloud through fog nodes. Fig.3 shows the computational overhead on data owners during this phase. We can see the encryption time of our scheme and Zhang et al. [32] is constant since most of the laborious decryption operations are delegated to the fog nodes, while it versus the number of attributes in access policy in Zuo et al. [29]. Fig.4 shows the decryption time on users versus the number of users' attributes. Specifically, in

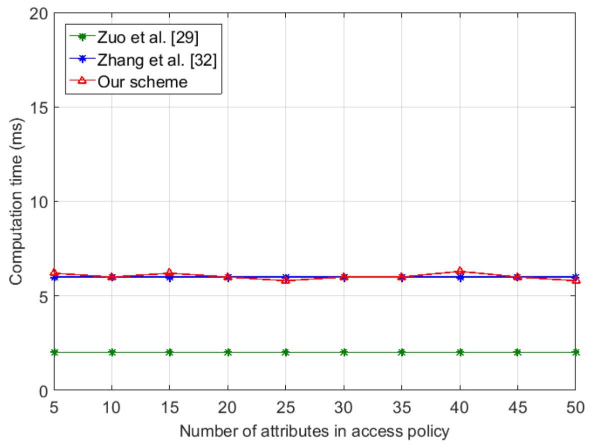


FIGURE 4. **Comparison of computational overhead for decryption.**

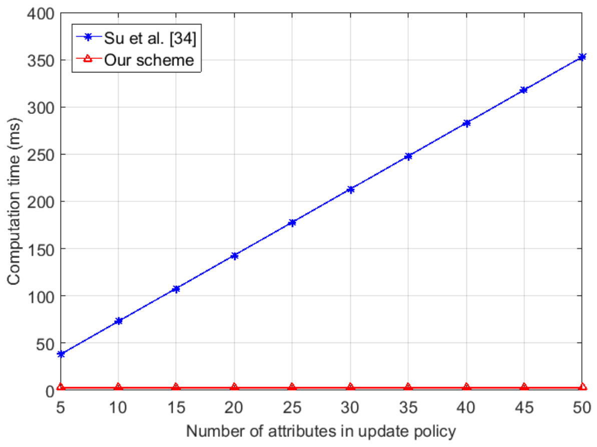


FIGURE 5. **Comparison of computational overhead for signing.**

Zuo et al. [29], Zhang et al. [32] and our scheme, the heavy computation operations of decryption are outsourced to external server, such as cloud servers and fog nodes, thus the computation operations for users to decrypt in these schemes are irrelevant to the number of attributes in the access policy.

The time complexity on users of ciphertext update which mainly refers to signing algorithms in both our scheme and Su et al. [34] is given in Fig.5. Concerning on the local computation performed by the signer, our scheme achieves much nearly constant performance compared with the linear increasing efficiency of the scheme of Su et al. [34] by outsourcing many computations to fog nodes. This advantage allows our scheme to be applied for the resource-constrained IoT devices to complete the signing task.

Moreover, we consider that the IoT device has limited storage ability. Since the outsourcing key can be firstly generated by attribute authority and then sent to the fog nodes. Therefore, the user only needs to store a small-sized component D locally but still maintaining encryption, decryption and signing capability. We argue that such amount is acceptable for IoT devices such as Samsung phone used in our experiments.

In summary, the experimental results show that our scheme incurs less computational cost on the encryption of data owner, the decryption and signing of user, which ensures both fine-grained data access control and efficient ciphertext update in fog computing. Hence, our scheme could be applied to smart healthcare, vehicular cloud computing, and etc.

VIII. CONCLUSION

In this paper, we propose a secure data access control scheme in fog computing for IoT based on CP-ABE and ABS. The sensitive data of users are first encrypted with both access policy and update policy, and then outsourced to cloud servers through fog nodes. Thus, the users whose attributes satisfy the access policy can decrypt the ciphertext. In order to address the issue of data modification, the CSP will check the signature, to ensure that only the users whose attributes satisfy the update policy can renew the ciphertext. Hence, our scheme achieves both fine-grained data access control and secure ciphertext update.

Moreover, our scheme presents an outsourced encryption, decryption and signing construction by delegating most of the operations to fog nodes. The extensive performance analysis and experiments are conducted, and the results indicate our scheme can well tolerate the increasing number of attributes, which is suitable for the resource-constrained IoT devices in fog computing.

REFERENCES

[1] C. Rong, S. T. Nguyen, and M. G. Jaatun, "Beyond lightning: A survey on security challenges in cloud computing," *Comput. Elect. Eng.*, vol. 39, no. 1, pp. 47–54, Jan. 2013.

[2] G. Fortino, A. Guerrieri, W. Russo, and C. Savaglio, "Integration of agent-based and cloud computing for the smart objects-oriented IoT," in *Proc. IEEE Int. Conf. Comput. Supported Cooperat. Work Design*, Hsinchu, Taiwan, May 2014, pp. 493–498.

[3] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of Things (IoT): A vision, architectural elements, and future directions," *Future Generat. Comput. Syst.*, vol. 29, no. 7, pp. 1645–1660, 2013.

[4] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the Internet of Things," in *Proc. 1st Ed. MCC Workshop Mobile Cloud Comput.*, Helsinki, Finland, 2012, pp. 13–16.

[5] P. Hu, H. Ning, T. Qiu, H. Song, Y. Wang, and X. Yao, "Security and privacy preservation scheme of face identification and resolution framework using fog computing in Internet of Things," *IEEE Internet Things J.*. [Online]. Available: https://doi.org/10.1109/JIOT.2017.2659783

[6] Q. Huang, Z. Ma, Y. Yang, J. Fu, and X. Niu, "EABDS: attribute-based secure data sharing with efficient revocation in cloud computing," *Chin. J. Electron.*, vol. 24, no. 4, pp. 862–868, 2015.

[7] Q. Huang, L. Wang, and Y. Yang, "DECENT: Secure and fine-grained data access control with policy updating for constrained IoT devices," in *World Wide Web*. May 2017. [Online]. Available: https://doi.org/10.1007/s11280-017-0462-0

[8] X. Dong, J. Yu, Y. Luo, Y. Chen, G. Xue, and M. Li, "Achieving secure and efficient data collaboration in cloud computing," in *Proc. IEEE/ACM 21st Int. Symp. Quality Service*, Montreal, QC, Canada, Jun. 2013, pp. 195–200.

[9] F. Zhao, T. Nishide, and K. Sakurai, "Realizing fine-grained and flexible access control to outsourced data with attribute-based cryptosystems," in *Proc. Inf. Secur. Pract. Exper.-7th Int. Conf.*, Guangzhou, China, 2011, pp. 83–97.

[10] J. Li, M. H. Au, W. Susilo, D. Xie, and K. Ren, "Attribute-based signature and its applications," in *Proc. 5th Int. Symp. Inf., Comput. Commun. Secur.*, Guangzhou, China, 2010, pp. 60–69.

[11] S. Ruj, M. Stojmenovic, and A. Nayak, "Decentralized access control with anonymous authentication of data stored in clouds," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 2, pp. 384–394, Feb. 2014.

[12] Q. Huang, Y. Yang, and M. Shen, "Secure and efficient data collaboration with hierarchical attribute-based encryption in cloud computing," *Future Generat. Comput. Syst.*, vol. 72, pp. 239–249, Jul. 2017.

[13] R. Lu, K. Heung, A. H. Lashkari, and A. A. Ghorbani, "A lightweight privacy-preserving data aggregation scheme for fog computing-enhanced IoT," *IEEE Access*, vol. 5, pp. 3302–3312, 2017.

[14] K. Lee, D. Kim, D. Ha, U. Rajput, and H. Oh, "On security and privacy issues of fog computing supported Internet of Things environment," in *Proc. Int. Conf. Netw. Future*, Montreal, QC, Canada, Sep. 2015, pp. 1–3.

[15] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *Proc. 24th Annu. Int. Conf. Theory Appl. Cryptograph. Techn.*, Aarhus, Denmark, 2005, pp. 457–473.

[16] A. Lewko and B. Waters, "Decentralizing attribute-based encryption," in *Proc. 30th Annu. Int. Conf. Theory Appl. Cryptograph. Techn.*, Tallinn, Estonia,2011, pp. 568–588.

[17] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *Proc. 13th ACM Conf. Comput. Commun. Secur.*, New York, NY, USA, 2006, pp. 89–98.

[18] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *Proc. IEEE Symp. Secur. Privacy*, Berkeley, CA, USA, Mays 2007, pp. 321–334.

[19] S. Ruj, A. Nayak, and I. Stojmenovic, "Distributed fine-grained access control in wireless sensor networks," in *Proc. IEEE Int. Parallel Distrib. Process. Symp.*, Anchorage, AK, USA, May 2011, pp. 352–362.

[20] S. Yu, K. Ren, and W. Lou, "FDAC: Toward fine-grained distributed data access control in wireless sensor networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 22, no. 4, pp. 673–686, Apr. 2011.

[21] C. Hu, H. Li, Y. Huo, T. Xiang, and X. Liao, "Secure and efficient data communication protocol for wireless body area networks," *IEEE Trans. Multi-Scale Comput. Syst.*, vol. 2, no. 2, pp. 94–107, Apr./Jun. 2016.

[22] Y. Jiang, W. Susilo, Y. Mu, and F. Guo, "Ciphertext-policy attribute-based encryption against key-delegation abuse in fog computing," *Future Generat. Comput. Syst.*, Jan. 2017. [Online]. Available: https://doi.org/10.1016/j.future.2017.01.026

[23] L.-Y. Yeh, P.-Y. Chiang, Y.-L. Tsai, and J.-L. Huang, "Cloud-based fine-grained health information access control framework for lightweight IoT devices with dynamic auditing and attribute revocation," *IEEE Trans. Cloud Comput.*, Oct. 2015. [Online]. Available: https://doi.org/10.1109/TCC.2015.2485199

[24] X. Chen, J. Li, X. Huang, J. Ma, and W. Lou, "New Publicly verifiable databases with efficient updates," *IEEE Trans. Depend. Sec. Comput.*, vol. 12, no. 5, pp. 546–556, Sep. 2015.

[25] X. Chen, J. Li, J. Ma, Q. Tang, and W. Lou, "New algorithms for secure outsourcing of modular exponentiations," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 9, pp. 2386–2396, Sep. 2014.

[26] S. Hohenberger and B. Waters, "Online/offline attribute-based encryption," in *Proc. 17th Int. Conf. Pract. Theory Public-Key Cryptogr.*, Buenos Aires, Argentina, 2014, pp. 293–310.

[27] N. Oualha and K. T. Nguyen, "Lightweight attribute-based encryption for the Internet of Things," in *Proc. 25th Int. Conf. Comput. Commun. Netw.*, Waikoloa, HI, USA, Aug. 2016, pp. 1–6.

[28] A. Lounis, A. Hadjidj, A. Bouabdallah, and Y. Challal, "Healing on the cloud: Secure cloud architecture for medical wireless sensor networks," *Future Generat. Comput. Syst.*, vol. 55, pp. 266–277, Feb. 2016.

[29] C. Zuo, J. Shao, G. Wei, M. Xie, and M. Ji, "CCA-secure ABE with outsourced decryption for fog computing," *Future Generat. Comput. Syst.*, Nov. 2016. [Online]. Available: https://doi.org/10.1016/j.future.2016.10.028

[30] Y. Yang, X. Zheng, and C. Tang, "Lightweight distributed secure data management system for health Internet of Things," *J. Netw. Comput. Appl.*, vol. 89, pp. 26–37, Nov. 2016. [Online]. Available: https://doi.org/10.1016/ j.jnca.2016.11.017

[31] L. Yang, A. Humayed, and F. Li, "A multi-cloud based privacy-preserving data publishing scheme for the Internet of Things," in *Proc. 32nd Annu. Comput. Secur. Appl. Conf.*, Los Angeles, CA, USA, 2016, pp. 30–39.

[32] P. Zhang, Z. Chen, J. K. Liu, K. Liang, and H. Liu, "An efficient access control scheme with outsourcing capability and attribute update for fog computing," *Future Generat. Comput. Syst.*, Dec. 2016. [Online]. Available: https://doi.org/10.1016/j.future.2016.12.015

[33] H. K. Maji, M. Prabhakaran, and M. Rosulek, "Attribute-based signatures," in *Proc. 11th Cryptographers' Track at RSA Conf., Topics Cryptol.*, San Francisco, CA, USA, 2011, pp. 376–392.

[34] J. Su, D. Cao, B. Zhao, X. Wang, and I. You, "ePASS: An expressive attribute-based signature scheme with privacy and an unforgeability guarantee for the Internet of Things," *Future Generat. Comput. Syst.*, vol. 33, no. 2, pp. 11–18, Apr. 2014.

[35] X. Chen, J. Li, X. Huang, J. Li, Y. Xiang, and D. S. Wong, "Secure outsourced attribute-based signatures," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 12, pp. 3285–3294, Dec. 2014.

[36] G. Fortino, A. Rovella, W. Russo, and C. Savaglio, "On the classification of cyberphysical smart objects in the Internet of Things," in *Proc. Int. Workshop Netw. Cooperating Objects Smart Cities*, Berlin, Germany, 2014, pp. 3–9.

[37] B. Lynn. *The Pairing-Based Cryptography Library*, accessed on 2013. [Online]. Available: http://crypto.stanford.edu/pbc/

QINLONG HUANG (M'17) received the Ph.D. degree from the School of Computer, Beijing University of Posts and Telecommunications, China, in 2014. He is currently an Associate Professor with the School of Cyberspace Security, Beijing University of Posts and Telecommunications, and an Associate Director of the National Engineering Laboratory for Disaster Backup and Recovery, China. He is also the Principal Investigator of the project funded by the National Natural Science Foundation of China. His research interests include cloud computing security, social network security, and IoT security. He was serving as a Reviewer of the IEEE Transactions on Information Forensics and Security, *ACM Computing Surveys*, the *ACM Transactions on Multimedia Computing, Communications and Applications*, *IET Information Security*, and *Security and Communication Networks*.

YIXIAN YANG received the Ph.D. degree from the Beijing University of Posts and Telecommunications, China, in 1988. He was a Changjiang Distinguished Professor of China in 1993. He is currently a Professor with the School of Cyberspace Security, Beijing University of Posts and Telecommunications, where he is also the Director of the National Engineering Laboratory for Disaster Backup and Recovery, Information Security Center. He has authored or co-authored over 300 journals and conference papers. His research interests include cryptography, information, and network security. He is a fellow of the China Institute of Communications and the Chinese Association for Cryptologic Research, and the Council Member of the Chinese Institute of Electronics. He was selected in National Science Fund for the Distinguished Young Scholars of China in 1994. He is the Editor-in-Chief of the *Journal on Communications*.

LICHENG WANG received the B.S. degree from Northwest Normal University, China, in 1995, the M.S. degree from Nanjing University, China, in 2001, and the Ph.D. degree from Shanghai Jiao Tong University, China, in 2007. He is currently an Associate Professor with the Beijing University of Posts and Telecommunications, China. His current research interests include applied cryptography and network security.