Efficient and Secure Attribute-Based Access Control With Identical Sub-Policies Frequently Used in Cloud Storage

Kaiping Xue, *Senior Member, IEEE*, Na Gai, Jianan Hong, David S.L. Wei, *Senior Member, IEEE*, Peilin Hong, and Nenghai Yu

Abstract—Under the assumption of honest-but-curious cloud service provider, various cryptographic techniques have been used to address the issues of data access control and confidentiality in public cloud storage. Among which, attribute-based encryption (ABE) has been shown to be an attractive scheme. Although the technique of ABE brings in various benefits, its onerous overhead should not be ignored. In this article, based on an improved LSSS (linear secret sharing scheme) matrix expression integrated in CP-ABE (Ciphertext-Policy Attribute-Based Encryption) algorithm, we present an efficient and secure attribute-based access control scheme for the scenarios where multiple data are shared and encrypted with frequently used sub-policies. In the scheme, a user can store the parameters about a specific sub-policy in his/her first decryption, which can be reused in the subsequent data decryptions whose embedded access policies include the same sub-policy so as to significantly reduce the computation cost. Our proposed scheme is proved to be semantically secure under chosen plaintext attacks and can well preserve the confidentiality of the data sharing system. Our analysis and experimentation also show that our scheme does significantly reduce the decryption time and while trades in only very little storage overhead, and thus effectively promotes the efficiency.

Index Terms—Cloud storage, access control, frequently used sub-policy, decryption promotion

1 Introduction

With rapid development of cloud computing and soaring requirement of large-volume data sharing, more and more individuals/corporations tend to outsource their data into the cloud [1], [2], [3], [4]. This is a win-win service paradigm for both cloud service client and service provider. For individuals/corporations, cloud storage provides them with convenient data sharing, reliable resource management in a pay-as-you-go manner or with long-term lease contracts [5], which is much more cost-effective compared with buying and maintaining the facilities by themselves [6]. Due to the advantage of centralized management, a cloud service provider can make more effective use of its powerful computation/storage resources and professional employees by renting various services out to the clients.

Apart from the benefits of cloud service, data confidentiality becomes a critical challenge: as data are stored by the honest-but-curious cloud service provider, it is not wise to rely on the service provider to execute data access control. The implementation of *Ciphertext-Policy Attribute-Based Encryption* (CP-ABE) [7] in cloud storage service solves the challenging issue of secure access control of outsourced data, and enables data owners to carry out fine-grained and flexible access control for their outsourced data, e.g., the work in [8], [9], [10], [11], [12], [13]. Based on CP-ABE [7], each user is issued with a secret key based on his/her attribute set, a file is encrypted under an access policy, and a user can decrypt the file if and only if his/her attribute set satisfies the access policy.

In this work, we focus on the scenarios where different access policies may contain identical sub-policies which are frequently used for data access. Such scenarios occur in quite a few organizations, enterprises, societies, etc., where a number of users access data for the purpose of a cooperated program, or for a common interesting topic. These users usually have a common attribute set, which comprises a social circle and can be expressed by an access sub-policy. Also, different files shared by an owner usually have relations among them in these scenarios. Let's take developing access policies for some important documents of a university as an example. Assume that some specific files can be accessed only by a user who is a PhD or a professor of the University of Science and Technology of China (USTC) and

- *K. Xue and N. Yu are with the Department of Electronic Engineering and Information Science, University of Science and Technology of China, Hefei 230027, China, and also with the School of Cyber Security, University of Science and Technology of China, Hefei, Anhui 230027, China. E-mail: {kpxue, ynh}@ustc.edu.cn.*
- *N. Gai is with the School of Cyber Security, University of Science and Technology of China, Hefei, Anhui 230027, China. E-mail: gn1996@mail.ustc.edu.cn.*
- *J. Hong is with Huawei Shanghai Research Institute, Shanghai 201206, China. E-mail: hongjianan@huawei.com.*
- *D. Wei is with Computer and Information Science Department, Fordham University, Bronx, NY 10458 USA. E-mail: wei@cis.fordham.edu.*
- *P. Hong is with Department of Electronic Engineering and Information Science, University of Science and Technology of China, Hefei, Anhui 230027, China. E-mail: plhong@ustc.edu.cn.*

Manuscript received 13 Mar. 2020; revised 6 Apr. 2020; accepted 11 Apr. 2020. Date of publication 15 Apr. 2020; date of current version 17 Jan. 2022. (Corresponding author: Kaiping Xue.)
Digital Object Identifier no. 10.1109/TDSC.2020.2987903

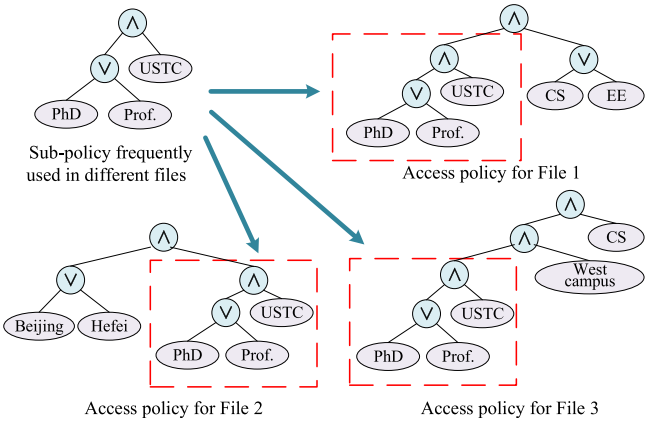


Fig. 1. Sub-policy embedded in multiple files.

his/her major is CS (Computer Science) or EE (Electronic Engineering), some files can be accessed only by a user who is a PhD or a professor from USTC and he/she works in Beijing or Hefei, while some other files can be accessed only by a user who is a PhD or a professor from USTC, majors in CS, and meanwhile he/she works in the west campus. Fig. 1 shows an instance in such scenario: assume that three files are outsourced to be stored in the cloud, and the corresponding access control policies are given in Fig. 1, respectively. These three policies all contain an identical sub-policy "'USTC' ∧ ('PhD' ∨ 'Prof.')", which is frequently used for different files. When using CP-ABE based access control schemes in such scenario, users would better execute the computation with the sub-policy for only once during multiple data decryption processes for different files. However, the existing CP-ABE based schemes is unable to achieve this goal.

To deal with the scenarios with frequently used sub-policies in cloud data sharing, we come up with a novel idea to design a new access control scheme with the following property: when a user queries one ciphertext and decrypts it, the decryption result not only recovers the currently accessed file, but also helps reduce the computation cost in the decryption procedure of subsequent ciphertexts embedded with identical sub-policies. Besides the requirement of efficiently reducing the computation overhead, the key challenge of our idea is how to provide the confidentiality preservation - our mechanism should not leave any opportunity for other users to get unauthorized information of the outsourced data.

In this paper, we thus devise an effective scheme to improve the performance of attribute-based data access control in the scenarios where multiple sets of shared data are encrypted with identical embedded sub-policies. More specifically, the first access of a file by a user saves the decrypted result with identical sub-policy, called identical sub-policy parameter, which can be reused to help significantly reduce the computation cost when conducting future decryption tasks for other files embedded with the same sub-policy.

In our design, users can store some decryption results in his/her first access, and then when he/she further queries other files embedded with the same sub-policy, only one-time pairing operation is required for the same sub-policy, if the file is published by the same data owner. In contrast, in the existing attribute-based access control, the number of expensive pairing operations is linear to the amount of required attributes to satisfy the sub-policy. Though, the main idea behind our design is simple, developing new encryption and decryption algorithms to realize it is quite challenging.

The main contributions of this paper can be summarized as follows:

1) We first propose an improved LSSS (linear secret sharing scheme) matrix expression to support different access policies with identical sub-policies, which can be integrated into CP-ABE algorithm.
2) Based on this expression method, we further presents a design of an attribute-based access control model to promote the efficiency of data decryption in the scenarios where multiple data are shared and encrypted with identical sub-policies in the cloud storage. In the scheme, a user can store the parameters about a specific sub-policy in his/her first decryption. This parameter can be reused in the subsequent data decryptions whose embedded access policies include the same sub-policy to significantly reduce the computation cost. Our further analysis shows that our design trades only little storage for saving significant computation cost.
3) We give a rigorous security proof to validate that the proposed scheme is semantically secure against chosen plaintext attacks. Meanwhile, it's proved that users cannot get any information from the unauthorized files individually or collusively.

The rest of this paper is organized as follows. In Section 2, we give the related works of data access control in public cloud. Then system model and security assumption are defined in Section 3, and preliminaries are reviewed in Section 4. We give the improved LSSS matrix expression in Section 5 and then present our access control scheme in Section 6. Section 7 analyzes the correctness, security, and performance properties. Finally, we conclude this paper in Section 8.

2 Related Work

Due to honest-but-curious cloud service provider, it is extremely challenging in protecting the security of outsourced data [14], [15], [16], [17]. Faced with the threat that cloud service provider may be curious about the stored information, many works have been proposed to preserve confidentiality of outsourced data, and to realize secure data access control [8], [9], [10], [12], [13], [18], [19], [20], [21]. Most of these works adopt ciphetext-policy attribute-based encryption (CP-ABE) [7] as the core cryptographic technique. In CP-ABE, each secret key is generated upon a set of attributes, and each file is encrypted with an embedded access policy. Intuitively, an access policy can be regarded as a tree, which well suits common access control models, e.g., RBAC (Role-based Access Control) [22].

When CP-ABE is utilized to conduct cloud data access control, the structure of Linear Secret Sharing Scheme (LSSS) matrix [23], [24], [25] is more efficient than the original structure [7]. Many works have been proposed to adjust CP-ABE algorithm to suit cloud architectures.

Multi-authority architectures [9], [10], [26], [27] have been proposed to address the practical deployment issue for cloud storage, where attributes are managed by different organizations. The threshold-based mechanism for authorities [28] makes it possible that users can legally obtain secret keys even when some authorities have broken down or been compromised. The implementation of traceability [29], [30] can prevent a semi-trusted authority from abusing its privilege to issue keys to unauthorized users. Data access policy updating [19], [31] and user revocation [32], [33] provide solutions for dynamic cloud access control systems.

However, the encryption and decryption processes of CP-ABE both involve multiple pairing operations, which introduce high computation cost [34], [35]. This becomes an obstacle for practical deployments, especially when users' devices are energy constrained. In order to reduce the computation burden of the clients (users or owners), mechanisms with lower cost have been proposed [9], [11], [36], [37], [38]. Online/offline encryption [37], [38] reduces the owner's computation burden during data publishing, as most encryption work has been prepared when the energy is sufficient. Decryption outsourcing [9], [11], [36], [39] can free up users' decryption burden. In these schemes, cloud service provider takes majority of decryption work, without getting any knowledge of data information. However, decryption outsourcing exposes users' attribute sets to the cloud, which induces user privacy concerns [40]. Some schemes, such as what proposed in [41], [42], have realized fast decryption for CP-ABE. However, in these two schemes, the decrease of decryption complexity largely sacrifices the storage overhead on users' secret keys and ciphertexts. For instance, in literature [42], in order to realize the fast decryption, an arbitrary access policy should be reorganized as a two-layer one, AND gate first and OR gate later. Briefly speaking, faced with the structure of Fig. 1b, the volume size will expand from 4 to 6 in [42]. If the access structure is more complicated, the ciphertext size will increase by many times. In KP-ABE based approach [41], the access policy is embedded in secret keys, which also increases the key size by many times.

Recently, Wang *et al.* [43] proposed a mechanism to encrypt multiple files in an integrated access structure. Different from other hierarchical ABE schemes, such as [8], [44], in [43], multiple files can be organized in a hierarchical way, which can significantly save the storage, encryption, and decryption cost with such integrated structure. However, the work of Wang *et al.* does not aim to handle data encryption under different access policies with identical sub-policies. Furthermore, based on the scheme in [43], all files should be outsourced to the cloud storage simultaneously.

Therefore, we are motivated to consider if there is an approach for owner and users to securely store some parameters in encryption and decryption such that repeated and complex computation can be avoided when multiple texts are encrypted with identical sub-policy.

3 System Model and Security Assumption

3.1 System Model

Our system model keeps consistent with the general system model of access control in the cloud storage scene. As depicted in Fig. 2, the data sharing system consists of four kinds of entities as follows:

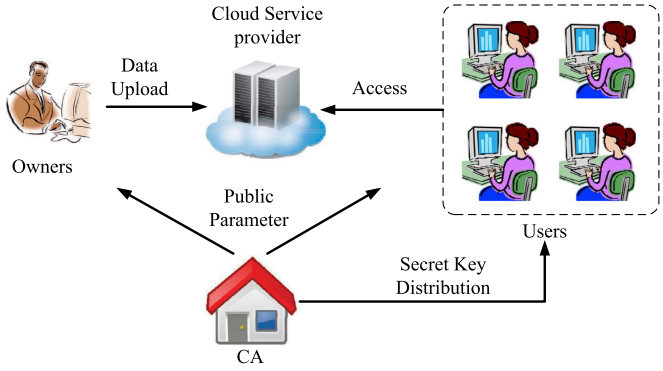


Fig. 2. System model.

- *Cloud service provider (cloud)* provides a storage platform and an interface for other entities to upload and download encrypted data. It does not conduct access control for stored data. We assume that the encrypted data can be downloaded freely by any data consumer, the same as the assumption in [9], [20], [28], [32].
- *The central authority (CA)* is responsible for managing the security protection of the shared data and their access control: it publishes system parameters and distributes secret keys related to specific attribute set for each user.
- *The data owner (owner)* stores and shares data in the cloud. In order to master the access control, each of the outsourced data should be encrypted under a designate access policy.
- *The data consumer (user)* is assigned with a secret key from CA. He/she can query any ciphertext stored in the cloud, but is able to decrypt it only if his/her attribute set satisfies the access policy.

3.2 Security Assumption

In our scheme, the cloud is assumed to be honest-but-curious [9], [10]. On the one hand, it offers a reliable storage service and correctly conducts all missions for other entities; On the other hand, it may try to gain unauthorized information for its own benefits.

CA is assumed to be fully trusted, which will issue secret keys to users according to their attribute sets strictly. The data owner will strictly define an access policy for his/her outsourced file and encrypt it under the access policy. In our design, the data owner is assumed to have a secure storage to securely maintain his/her parameters.

We assume that some malicious users exist in the system to try to decrypt any ciphertexts to obtain unauthorized data by all means, including colluding with other users.

4 Preliminaries

4.1 Bilinear Pairings

Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be two multiplicative cyclic groups with the same prime order p. Let $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ be a bilinear map holding the following properties:

1) *Bilinearity.* For all $u, v \in \mathbb{G}_1$ and $a, b \in \mathbb{Z}_p$, we have $e(u^a, v^b) = e(u, v)^{ab}$.
2) *Non-degeneracy.* If g is a generator of $\mathbb{G}_1$, then $e(g, g)$ is also a generator of $\mathbb{G}_2$.
3) *Computability.* There is an efficient algorithm to compute $e(u, v)$ for all $u, v \in \mathbb{G}_1$.

Definition 1 (Decisional Parallel Bilinear Diffie-Hellman Exponent Assumption) [23]. *The decisional* q parallel-*BDHE assumption is defined as follows. Let a, s, $b_1, \ldots, b_q \in_R \mathbb{Z}_p$, and g be a generator of $\mathbb{G}_1$. Given the information $\vec{y}$ as*

$$g, g^s, g^a, \ldots, g^{a^q};\ g^{a^{q+2}}, \ldots, g^{a^{2q}}$$
$$\forall 1 \le j \le q : g^{s \cdot b_j}, g^{a/b_j}, \ldots, g^{a^q/b_j};\ g^{a^{q+2}/b_j}, \ldots, g^{a^{2q}/b_j},$$
$$\forall 1 \le j,\ k \le q, k \ne j :\ g^{a \cdot s \cdot b_k/b_j}, \ldots, g^{a^q \cdot s \cdot b_k/b_j},$$

there is a probabilistic polynomial-time adversary $\mathcal{A}$ to distinguish $e(g, g)^{a^{q+1}s}$ from a random element $Z = \mathbb{G}_2$. If the advantage of the adversary $\mathcal{A}$ is negligible, which is defined as follows:

$$\left| Pr[\mathcal{A}(\vec{y}, e(g, g)^{a^{q+1}s}) = 0] - Pr[\mathcal{A}(\vec{y}, Z) = 0] \right| < \epsilon, \tag{1}$$

we can say that the scheme is secure.

For more details about bilinear pairings and its applications, interesting readers can refer to [23], [45], [46].

4.2 Security Model

The security model is formalized by the following security game between an adversary $\mathcal{A}$ and a challenger $\mathcal{C}$.

- *Setup.* The challenger $\mathcal{C}$ takes a security parameter λ to run the *setup* algorithm and gives the public parameters PK to $\mathcal{A}$, and keeps the master key MSK secret.
- *Phase 1.* $\mathcal{A}$ makes a secret key request according to an arbitrary attribute set S_1, and $\mathcal{C}$ gives the corresponding answer with the secret key SK to $\mathcal{A}$.
- *Challenge.* $\mathcal{A}$ submits two equal-length files $\mathcal{M}_0$ and $\mathcal{M}_1$ to $\mathcal{C}$, with an access policy (M, ρ), which cannot be satisfied by attribute set S_1. $\mathcal{C}$ randomly selects $\mathcal{M}_\nu$, $\nu \in (0, 1)$ and encrypts it under (M, ρ). Here, the access policy (M, ρ) follows the structure of LSSS matrix, which is to be introduced in the next section.
- *Phase 2.* Repeat *Phase 1* with an attribute set S_2, where $S_1 \bigcup S_2$ cannot satisfy (M, ρ).
- *Guess.* $\mathcal{A}$ outputs a guess ν' of ν. The advantage of $\mathcal{A}$ is defined as:

$$p = \left| Pr[\nu' = \nu] - \frac{1}{2} \right|. \tag{2}$$

Definition 2. *Similar to the scheme in [47], the proposed scheme is secure against the chosen plaintext attack if all probabilistic polynomial-time adversaries have at most a negligible advantage in the above game, i.e., $p < \epsilon$.*

4.3 Access Structure and Linear Secret Sharing Scheme

Definition 3 (Access Structure). *Let $\mathcal{P} = \{P_1, P_2, \ldots, P_n\}$ be a set of parties. A collection $\mathbb{A} \subseteq 2^{\{P_1, P_2, \ldots, P_n\}}$ is monotonic if $\forall$ $\mathbf{B}$, $\mathbf{C}$: if $\mathbf{B} \in \mathbb{A}$ and $\mathbf{B} \subseteq \mathbf{C}$, then $\mathbf{C} \in \mathbb{A}$. An access structure (respectively, monotonic access structure) is a collection (respectively, monotonic collection) $\mathbb{A}$ of non-empty subsets of $\mathcal{P}$, i.e., $\mathbb{A} \subseteq 2^{\{P_1, P_2, \ldots, P_n\}} \backslash \{\emptyset\}$. The sets in $\mathbb{A}$ are called authorized ones, and the sets not in $\mathbb{A}$ are called unauthorized ones.*

TABLE 1
Policy Illustrations

Notation	Description	Remarks
l	# of attributes in policy	
l_s	# of attributes in sub-policy	$l = l_s + l_1 + l_2$
n	dimensions spanned by the policy	
n_s	dimensions spanned by the sub-policy	$n = n_s + n_1$

Observing the constructions in [7], [23], [47], an LSSS access structure can be used to denote the access policy $\mathbb{A}$. Following the method defined in [48], any monotonic boolean formula can be converted into an LSSS representation. The description of LSSS is presented in what follows.

Definition 4 (Linear Secret Sharing Scheme (LSSS) [23]). *There is a secret sharing scheme with a sharing-generating matrix $M^{l \times n}$ with l rows and n columns. Let $\rho(i)$ be the party that labels the ith row of M. Choose a column vector $\overrightarrow{v} = (s_0, r_2, \ldots, r_n) \in_R \mathbb{Z}_p^n$, and $\lambda = (M \cdot \overrightarrow{v})$ is the vector of l shares of the secret s_0. The ith dimension of $\overrightarrow{\lambda}$, denoted as λ_i, is the secret share belonging to party $\rho(i)$.*

Let $I \subset \{1, 2, \ldots, l\}$. If $\{\rho(i) : i \in I\}$ is an attribute set that satisfies the access policy (M, ρ), then there exist constants $\{\omega_i \in \mathbb{Z}_p\}_{i \in I}$ such that $\sum_{i \in I} \omega_i \lambda_i = s_0$. These constants $\{\omega_i\}$ can be found in time polynomial in the size of the share-generating matrix $M^{l \times n}$. Note that, for any unauthorized set $\mathcal{S} \notin \mathbb{A}$, no such constants $\{\omega_i\}$ exist.

5 Improved LSSS Matrix Expression to Support Identical Sub-Policy in Different Access Policies

In our work, we focus on the access policies that contain identical sub-policy. Each of the policies can be expressed by a matrix as the following format:

$$\begin{pmatrix} M_1^{l_1 \times n_1} & 0^{l_1 \times n_s} \\ \hline \overrightarrow{m_s}^{1 \times n_1} & \\ \vdots & M_s^{l_s \times n_s} \\ \overrightarrow{m_s}^{1 \times n_1} & \\ \hline M_2^{l_2 \times n_1} & 0^{l_2 \times n_s} \end{pmatrix}, \tag{3}$$

where there are l_s duplicated row vector m_s on the left side of the submatrix M_s. Table 1 illustrates some parameters for matrix M, and all blocks in M hold the following properties:

- M_s: If a frequently accessed sub-policy is regarded as an individual access policy, the relevant matrix is as:

$$M_{sub} = \begin{pmatrix} 1^{l_s \times 1} & M_s \end{pmatrix}, \tag{4}$$

where $1^{l_s \times 1}$ is an all-one column vector. The row size l_s indicates the total number of attributes in the sub-policy, and n_s is the dimension spanned by the sub-policy.

- Other blocks $(M_1, \overrightarrow{m_s}, M_2)$: If the sub-policy is regarded as one single attribute of the access policy (or one leaf node in the tree), this policy can be expressed by a new matrix:

$$M' = \begin{pmatrix} M_1 \\ \overrightarrow{m_s} \\ M_2 \end{pmatrix}, \tag{5}$$

where M' is a matrix of $l_1 + 1 + l_2$ rows and n_1 columns. Especially, $\overrightarrow{m_s}$ is the row vector labelling the sub-policy to satisfy the linear secret sharing requirement of the new policy, where the sub-policy is manually set as one single node.

It is worth noting that matrix M should obey some rules due to the format in Eq. (3). An intuitive expression can be described as: only the users satisfying sub-policy M_s may be able to satisfy M. Also, this work does not take into account the case where M_s is only an optional requirement for M for the following considerations:

1) As we talk about the scenarios such as data sharing among people with common interests or in the same group, users accessing these data should all be in a common social circle described by a sub-policy. Thus, users who do not satisfy the sub-policy are those who do not belong to the same group.
2) This feature gives us the potential to optimize encryption/decryption performance. To present a mechanism from having totally arbitrary policy without any constraint is somewhat too challenging and impractical either. In our research, we have put forward relevant solutions, but the efficiency promotion becomes less significant. Thus, we found that the restrictions from real world has have their own values in our design.

In fact, there is a special category of CP-ABE based researches which only consider AND gates for access policies, e.g., [49], [50], [51], where AND gate is the only logic to construct the access policy in practical usage. The access policy in our scheme is more general than that of those schemes. The special requirement in our scheme is that the frequently used sub-policies in the form of a sub-tree should be integrated to the policy tree via AND gate, which is acting as an attribute and must be satisfied by potential users. Note that based on the above discussion, we believe that the policy restriction of our current work does not affect the functionality too much.

6 Our Access Control Scheme for Data Sets with Frequently Used Sub-Policy

6.1 Overview of Our Scheme

In order to increase the decryption efficiency, where different data sets are embedded with an identical sub-policy, we present an attribute-based access control model, as shown in Fig. 3. Considering a scenario that an owner outsources multiple files and the access privileges of the files are released to different sets of users embedded with an identical sub-policy. When the owner first executes the encryption algorithm, besides CT, an identical sub-policy parameter can also be securely stored in his/her device. This parameter can

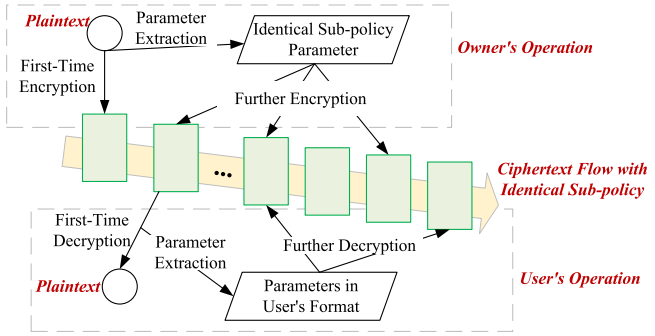


Fig. 3. The usage of identical sub-policy parameter: It is generated in the first encryption/decryption, and assist further encryption/decryption.

be utilized later to assist the execution of subsequent encryptions. For the user, the decryption algorithm will also be executed for many times. When he/she first accesses one of these files, the decryption outputs not only the plaintext, but also the relevant identical sub-policy parameter stored in his/her device. In the future, when the data being accessed are with the same sub-policy and published by the same owner, the saved parameters can be reused to help the user to skip sub-policy related operations, thereby significantly promoting the decryption efficiency.

The basic requirement is that the use of identical sub-policy parameters should be devised in such a way that the confidentiality property is still maintained and will not sacrifice too much other type of performance. After the detailed construction description, we will analyze these features thoroughly.

6.2 Construction

6.2.1 Setup

CA chooses two multiplicative cyclic groups $\mathbb{G}_1$ and $\mathbb{G}_2$ with the same prime order p, and defines a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$. CA also selects a generator $g \in \mathbb{G}_1$, and a hash function $H : (0,1)^* \rightarrow \mathbb{G}_1$. It further chooses two random secrets $\alpha, a \in \mathbb{Z}_p$. The public parameter is published as

$$PK = \{p, \mathbb{G}_1, \mathbb{G}_2, g, e, H, e(g,g)^{\alpha}, g^{a}\}.$$

Then, CA securely stores its master secret key as $MSK = g^{\alpha}$.

6.2.2 Encryption

Before uploading shared file $\mathcal{M}$, the data owner should encrypt it under the designed access policy (M, ρ). He/she first uses a symmetric cryptography to encrypt data $\mathcal{M}$ with a randomly chosen key $\mathcal{K} \in \mathbb{G}_2$.

Let M be a matrix in the form of Eq. (3). The encryption algorithm differs a little, which depends on whether it is executed first or subsequently by the data owner, or otherwise.

If it is the first time to encrypt a file with a frequent sub-policy, the procedure is given as follows:

A vector is set as $\overrightarrow{v} = (s_0, r_2, \ldots, r_{n_1}, y_1, \ldots, y_{n_s}) \in_R \mathbb{Z}_p^{n_1+n_s}$. The owner computes $M\overrightarrow{v}$. For $i \in (1, l_1 + l_s + l_2)$, $\lambda_i = (M\overrightarrow{v})_i$, denoted as the ith dimension of $M\overrightarrow{v}$; and $r_i \in_R \mathbb{Z}_p^*$. The ciphertext is generated as:

$$CT = \{\hat{C} = E_{\mathcal{K}}(\mathcal{M}), C = \mathcal{K}e(g,g)^{\alpha s_0}, C' = g^{s_0},$$
$$\forall i \in (1, l_1 + l_s + l_n) : C_i = g^{a\lambda_i} H(\rho(i))^{-r_i}, C_i' = g^{r_i}\}.$$

TABLE 2
user's Identical Sub-Policy Parameter

Sub-Policy	Sequence ID	sub-policy parameter
$(M_s, \rho_s)_1$	$seq_s(1)$	$(F^*_{i_0}, F^*_{sub})_1$
	$seq_s(2)$	$(F^*_{i_0}, F^*_{sub})_2$
$(M_s, \rho_s)_2$	$seq_s(3)$	$(F^*_{i_0}, F^*_{sub})_3$

The uploaded file is in an encapsulated format as follows,

$$(M, \rho) \mid (M_s, \rho_s), seq_s \mid CT,$$

where (M, ρ) denotes the access policy of the file; (M_s, ρ_s) denotes the designate sub-policy ($\rho_s(i) = \rho(i + l_1)$, l_1 is the row number of M_1 in Eq. (3)), and seq_s can be regarded as an identity (ID) to distinguish different owners for the identical sub-policy. The data owner securely stores the identical sub-policy parameter of data owner as:

$$\{(M_s, \rho_s), seq_s, Y = (y_1, \ldots, y_{n_s})\}.$$

If it is not the first time to encrypt the file embedded with the same sub-policy, then there exists the relevant parameter in the owner's device. The generation of $\overrightarrow{v}$ is different: randomly choose $\overrightarrow{v_0} = (s_0, r_2, \ldots, r_{n_1})$. The vector is a concatenation of $\overrightarrow{v_0}$ and Y, where Y is the content of the parameter. The remaining steps proceed in the same way as the first encryption.

6.2.3 KeyGen

For each user U_j with attribute set S_j, CA first randomly chooses $u_j \in \mathbb{Z}^*_p$ as a unique identity for the user. Then, CA computes the user's secret key as:

$$SK = \{K = g^{\alpha} g^{au_j}, L = g^{u_j}, \forall x \in S_j : K_x = H(x)^{u_j}\}.$$

At the end of this procedure, SK_j is sent to U_j in a secure tunnel. Furthermore, u_j, as a secret parameter to resist potential collusion attacks, can be stored securely for the future key requests of extra attributes by U_j such that the user's attribute associated components will be linked with the same u_j, even if obtained at different times of key distributions. Compared with the entire secret key SK, the volume of "u_j"s is indeed very small. It's important to note that similar method is also adopted in many other related multi-authority CP-ABE schemes, e.g., in literature [9]. Thus, we think, it is an acceptable and recognized solution for our scheme.

6.2.4 Decryption

A user stores Table 2 to maintain user's identical sub-policy parameter. When getting CT, he/she first looks up Table 2 to see if the sub-policy and sequence ID is stored.

If not stored (first-time access), the algorithm proceeds as follows: Suppose $S^{sub}_j \subset S_j$ satisfies the sub-policy (M_{sub}, ρ_s) where M_{sub} is composed as Eq. (4). Let $I_s \subset \{1, 2, \ldots, l_s\}$ be defined as $I_s = \{i : \rho_s(i) \in S^{sub}_j\}$. If I_s satisfies M_{sub}, the user computes $\{\omega'_i\}_{i \in I_s}$ to solve the following equation:

$$\sum_{i \in I_s} \omega'_i M_{sub}(i, j) = \begin{cases} 1, & j = 1, \\ 0, & 2 \le j \le n_s. \end{cases} \tag{6}$$

The user arbitrarily selects $i_0 \in I_s$ and computes:

$$F_{i_0} = e(C_{i_0}, L) e(C'_i, K_{\rho_s(i_0)}) = e(g, g)^{u_j a \lambda_{i_0 + l_1}}, \tag{7a}$$

$$F_{sub} = \prod_{i \in I_s} (e(C_i, L) e(C'_i, K_{\rho_s(i)}))^{\omega'_i}. \tag{7b}$$

The generated F_{sub} can be regarded as $e(g, g)^{u_j a \lambda_{sub}}$, where λ_{sub} is regarded as the secret share for the sub-policy. The computation of F_{sub}, as illustrated above, is called *sub-policy related decryption* in our scheme.

Now the user gets M' using Eq. (5), and maps ρ to ρ' using the following rules:

$$\rho'(i) = \begin{cases} \rho(i), & 1 \le i \le l_1, \\ sub - policy, & i = l_1 + 1, \\ \rho(i + l_s - 1), & l_1 + 2 \le i \le l_1 + 1 + l_2. \end{cases}$$

Let $I \subset \{1, \ldots, l_1 + 1 + l_2\} - \{l_1 + 1\}$ be defined as $I = \{i : \rho'(i) \in S_j\}$. If $I \bigcup \{l_1 + 1\}$ satisfies M', the user computes $\{\omega_i\}_{i \in I \bigcup \{l_1 + 1\}}$ to hold:

$$\sum_{i \in I \bigcup \{l_1 + 1\}} \omega_i M'(i, j) = \begin{cases} 1, & j = 1, \\ 0, & 2 \le j \le n_1. \end{cases} \tag{8}$$

Let λ'_i be computed using λ_i in the same way as obtaining $\rho'(i)$, the user computes:

$$\begin{aligned} \mathcal{K}' &= C / \frac{e(C', K)}{F^{\omega_{l_1+1}}_{sub} \prod_{i \in I} (e(C_i, L) e(C'_i, K_{\rho'(i)}))^{\omega_i}} \\ &= \frac{\mathcal{K} e(g, g)^{\alpha s_0} \cdot e(g, g)^{u_j a (\omega_{l_1+1} \lambda_{sub} + \sum_{i \in I} \omega_i \lambda'_i)}}{e(g, g)^{\alpha s_0} e(g, g)^{u_j a s_0}}. \end{aligned} \tag{9}$$

Finally, the recovery of the file is as:

$$\mathcal{M} = \quad Dec_{\mathcal{K}'}(\hat{C}).$$

Let $F^*_{i_0} = F_{i_0}$ and $F^*_{sub} = F_{sub}$, and the user stores these components with the sub-policy and sequence ID in Table 2.

If the relevant sub-policy and sequence ID is already stored in the table, the *sub-policy related decryption* is more efficient: In the sub-policy (M_{sub}, ρ_s), the user selects the same i_0 and computes F_{i_0} as Eq. (7a), and F_{sub} is computed as:

$$F_{sub} = F^*_{sub} \cdot F_{i_0} / F^*_{i_0}. \tag{10}$$

The remaining steps are the same as the procedure without user's parameter, and the computed F_{sub} is used to output $\mathcal{K}$ in Eq. (9), which is to be proved in Section 7.1. It is worth noting that our scheme is compatible with the situation where there is no substructure in the LSSS-compatible access policy. Under this circumstance, encryption is conducted according to the first case in the text. The only difference is that the common sub-policy is not defined and mentioned. Thus, no relevant intermediate parameters should be included. Accordingly, for users, no relevant intermediate parameters should be cached and the decryption should be implemented using standard CP-ABE algorithm.

6.3 Necessary Additions When Considering Revocation

Our proposed scheme can also introduce the same revocation mechanisms in some related CP-ABE based work, such as the work of [9], without reducing the degree of confidentiality for most cases except one special case – When a user whose attribute set used to satisfy the sub-policy is revoked. In this scenario, the sub-policy is no longer satisfied.

In this section, we do not address ciphertext re-encryption and secret key update for user revocation, but only introduce our proposed procedure for sub-policy related executions.

The data owner and nonrevoked users separately conduct the following procedures:

- *Owner.* In the first encryption after a relevant revocation, the owner re-selects the vector $Y = (y_1, \ldots, y_{n_s})$ as a new-version of the parameter for the sub-policy.
- *Nonrevoked User.* When a user receives a ciphertext in a new version (the version update information can be acknowledged by some means, e.g., notified along with seq_s), the user decrypts it as the first-time decryption and follows Eqs. (7a) and (7b), respectively, to compute F_{i_0} and F_{sub} as his/her parameter's new-version for this sub-policy.

7 Correctness, Security, and Performance Analysis

7.1 Correctness

The correctness of our scheme can be analyzed in terms of two cases: the decrypt process without user's stored identical sub-policy parameter, and the process with user's stored identical sub-policy parameter.

To prove the correctness of decryption without user's stored parameter, we first define a notation $\hat{\omega}_i$ as:

$$\hat{\omega}_i = \begin{cases} \omega_i, & i \le l_1, \\ \omega'_{i-l_1} \cdot \omega_{l_1+1}, & l_1 < i \le l_1 + l_s, \\ \omega_{i-l_s+1}, & i > l_1 + l_s. \end{cases} \tag{11}$$

where $\{\omega_i\}$ and $\{\omega'_i\}$ are the constants whose relevant attributes satisfy the policy of M' and M_{sub}, respectively. Let I_0 be the index set of $\hat{\omega}_i$, and $I'_s \subset I_0$ be the index set whose elements label the attributes of the sub-policy. For $1 \le j \le n_1$,

$$\begin{aligned} \sum_{i \in I_0} \hat{\omega}_i M(i,j) &= \sum_{i \in I'_s} \hat{\omega}_i M(i,j) + \sum_{i \in I_0 - I'_s} \hat{\omega}_i M(i,j) \\ &= \omega_{l_1+1} \overrightarrow{m_s}(j) \sum_{i \in I_s} \omega'_i + \sum_{i \in I} \omega_i M'(i,j). \end{aligned} \tag{12}$$

We can compute the subtraction of Eq. (8) from Eq. (12) as:

$$Eq.(12) - Eq.(8) = \left(\textstyle\sum_{i \in I_s} \omega'_i - 1\right) \cdot \omega_{l_1+1} \overrightarrow{m_s}(j) = 0,$$

where $\sum_{i \in I_s} \omega'_i = 1$ holds because of Eq. (6) together with the format of M_{sub} defined in Eq. (4). Moreover, for $n_1 < j \le n_1 + n_s$, we have:

$$\begin{aligned} \sum_{i \in I_0} \hat{\omega}_i M(i,j) &= \sum_{i \in I'_s} \hat{\omega}_i M(i,j) + \sum_{i \in I_0 - I'_s} \hat{\omega}_i M(i,j) \\ &= \omega_{l_1+1} \sum_{i \in I_s} \omega'_i M_{sub}(i, j - n_1 + 1). \end{aligned} \tag{13}$$

According to Eq. (6), the above equation outputs 0 for all j. We summarize Eqs. (12) and (13) as:

$$\sum_{i \in I_0} \hat{\omega}_i M(i,j) = \begin{cases} 1, & j = 1, \\ 0, & \text{otherwise.} \end{cases} \tag{14}$$

As $\lambda_i = (M\overrightarrow{v})_i$, we can further have: $\sum_{i \in I_0} \hat{\omega}_i \lambda_i = s_0$, which equals to $(\omega_{l_1+1}\lambda_{sub} + \sum_{i \in I} \omega_i \lambda'_i)$ in Eq. (9). The analysis shows that $\mathcal{K}' = \mathcal{K}$, which can correctly decrypt $\mathcal{M}$ with a symmetric cryptographic algorithm.

When encountering a decryption process with user's identical sub-policy parameter, we prove the correctness of decryption as follows. We compare the secret shares of two files $\{\lambda_i\}$ with the same sub-policy (denoted as λ_i^{new} and λ_i^{old}). As the data owner uses the same M_s and Y in the two files, if s is denoted as r_1, then for each $i - l_1 \in I_s$ (for the clarity of analysis, we assume that $l_1^{old} = l_1^{new}$) we have:

$$\lambda_i^{new} - \lambda_i^{old} = \textstyle\sum_{j=1}^{n_1} (M^{new}(i,j) \cdot r_j^{new} - M^{old}(i,j) \cdot r_j^{old}),$$

where in M, for $l_1 < i, i' \le l_1 + l_s$, we have $M(i,j) = M(i',j)$. Then, $\lambda_i^{new} - \lambda_i^{old}$ becomes a constant for any attribute in the sub-policy. From this perspective, we have:

$$\lambda_{sub}^{new} - \lambda_{sub}^{old} = (\lambda_{i_0}^{new} - \lambda_{i_0}^{old}) \sum_{i \in I_s} \omega'_i, \quad \forall i_0 \in I_s.$$

This proves the correctness of Eq. (10), which is the result of *sub-policy related decryption*. As it is the only difference between the two cases, the correctness of this case is proved.

7.2 Security Analysis

1) Fine-Grained Access Control: The proposed scheme provides data owner with the capability to define an arbitrary access policy. With the access policy embedded in the ciphertext, a user can decrypt the ciphertext to access the data, only if his/her attribute set satisfies the policy. As shown in Eqs. (6) and (8), the constants $\{\omega'_i\}$ and $\{\omega_i\}$ exist only when the attribute set satisfies the sub-policy and entire access policy, respectively.

Especially, when a user has already stored F^*_{sub} for the frequently used sub-policy, it means that he/she must have an attribute set to satisfy the sub-policy, although only one attribute is used to proceed current computation. Thus, the security properties are preserved in our mechanism.

Another kind of adversaries, who have partial attributes for the sub-policy, but do not satisfy this sub-policy, are also in our considerations. In this case, the adversary will not be able to guess out F_{sub} due to the lack of attributes. A more strict proof is given in Section 7.2 3.

2) Security against Collusion Attack: Each user's attribute-related secret key K_x is made unknown to any other one by a secret number $u_j \in \mathbb{Z}_p^*$. Thus, it is impossible for two or more users to collude and decrypt the ciphertext, if none of them is able to decrypt it individually. Moreover, sub-policy parameter $(F^*_{i_0}, F^*_{sub})$ is also generated with secret number u_j. Thus, the identical sub-policy parameter of one user cannot be used by any other users.

From this perspective, an adversary who obtains multiple secret keys (each belongs to a forged identity) can also be resisted, as long as none of his/her single secret key's associated attribute sets can satisfy the access policy. Note

that if the colluded users are regarded as an entire adversary, these two attack models are still the same on the security aspect.

3) Data Confidentiality:

Theorem 1. *Suppose that decisional* q-parallel *BDHE assumption holds, then no probabilistic polynomial-time adversary can break the proposed scheme, with a challenge access policy (M^*, ρ^*), where M^* is an $l^* \times n^*$ matrix, and $l^*, n^* \leq q$.*

Proof. Assume that a probabilistic polynomial-time adversary $\mathcal{A}$ can compromise our scheme with advantage ϵ. We build a simulator $\mathcal{B}$ that can play decisional *q-parallel* BDHE assumption with advantage ϵ as follows.

Setup. First, $\mathcal{B}$ takes in a *q-parallel* BDHE challenge $(\vec{y}, T)$, where T equals to $e(g,g)^{a^{q+1}s_0}$ or a random $Z \in \mathbb{G}_2$, each with probability 0.5. $\mathcal{B}$ randomly selects $\alpha' \in \mathbb{Z}_p$ and computes $e(g,g)^{\alpha} = e(g^a, g^{a^q})e(g,g)^{\alpha'}$, such that α equals to $\alpha' + a^{q+1}$. Then, all components of PK are generated, except the function H, which is simulated by a random oracle in the next phase.

Phase 1. $\mathcal{A}$ gives a challenge access policy (M^*, ρ^*) and a secret key request according to an attribute set S_1, where S_1 does not satisfy (M^*, ρ^*). $\mathcal{B}$ first sets the random oracle H by building a table. Consider a call to $H(x)$, if it is already defined in the table, the oracle returns it. Otherwise, choose a random value z_x. Let X be the set of all row indices i such that $\rho^*(i) = x$, meaning that these rows match the same attribute x. The oracle is programmed as:

$$H(x) = g^{z_x} \prod_{i \in X} g^{aM^*_{i,1}/b_i} \cdot g^{aM^*_{i,2}/b_i} \cdots g^{aM^*_{i,n^*}/b_i}.$$

Note that if x does not exist in the policy, then $X = \varnothing$, and $H(x) = g^{z_x}$. Also, the distribution of $H(x)$ is random due to the g^{z_x} value.

To answer the key request, $\mathcal{B}$ first chooses $r \in_R \mathbb{Z}_p$, and a vector $\vec{\omega} = (\omega_1, \ldots, \omega_{n^*}) \in \mathbb{Z}_p^{n^*}$ such that $\omega_1 = -1$, and for all i such that $\rho^*(i) \in S_1$, we have $\vec{\omega} \cdot M^*_i = 0$. Note that $\vec{\omega}$ must exist. Then $\mathcal{B}$ computes L as

$$L = g^r \prod_{i=1,\ldots,n^*} (g^{a^{q+1-i}})^{\omega_i}.$$

Also, L can be defined as g^t, where t implicitly equals to $r + \sum_{i=1}^{n^*} \omega_i a^{q-i+1}$. $\mathcal{B}$ computes K as:

$$K = g^{\alpha'} g^{ar} \prod_{i=2}^{n^*} (g^{a^{q+2-i}})^{\omega_i}.$$

To generate K_x, $\forall x \in S_1$, $\mathcal{B}$ executes what follows. If there is no i such that $\rho^*(i) = x$, we compute $K_x = K^{z_x}$. Otherwise, we use the same notation X to represent the set of row indices that $\rho^*(i) = x$. K_x is generated as:

$$K_x = L^{z_x} \prod_{i \in X} \prod_{j=1}^{n^*} \left(g^{(a^j/b_i)r} \prod_{k=1, k \neq j}^{n^*} (g^{a^{q+1+j-k}/b_i})^{\omega_k} \right)^{M^*_{i,j}}.$$

Challenge. $\mathcal{A}$ gives the two files $\mathcal{M}_0$ and $\mathcal{M}_1$ to $\mathcal{B}$. $\mathcal{B}$ flips a coin $\nu \in (0,1)$ and creates $\hat{C} = E_{\mathcal{K}}(\mathcal{M}_\nu)$, $C = \mathcal{K}T \cdot e(g^{s_0}, g^{\alpha'})$, and $C' = g^{s_0}$, where $\mathcal{K} \in_R \mathbb{G}_2$. Then, $\mathcal{B}$ chooses random numbers $y'_2, \ldots, y'_{n^*}$ and the share of secret using vector

$$\vec{v} = (s_0, s_0 a + y'_2, s_0 a^2 + y'_3, \ldots, s_0 a^{n-1} + y'_{n^*}).$$

It additionally chooses random numbers $r'_1, \ldots r'_{l^*}$. For each row in M^*, we define R_i as the set of all other rows, k, such that $\rho^*(i) = \rho^*(k)$. The relevant components are generated as:

$$C_i = H(\rho^i)^{r'_i} \Big(\prod_{j=2}^{n^*} (g^a)^{M^*_{i,j} y'_j} \Big) (g^{b_i \cdot s_0})^{-z_{\rho^*(i)}} \cdot \prod_{k \in R_i} \prod_{j=1}^{n^*} (g^{a^j s_0 (b_i/b_k)})^{M^*_{k,j}},$$

$$C'_i = g^{-r'_i} g^{-s_0 b_i}.$$

Phase 2. Repeat *Phase 1* with the requested attribute set S_2.

Guess. $\mathcal{A}$ submits a guess ν' of ν. If $\nu' = \nu$, $\mathcal{B}$ then outputs $T = e(g,g)^{a^{q+1}s_0}$; otherwise, it outputs T which is a random number $Z \in \mathbb{G}_2$. With this policy, $\mathcal{B}$'s advantage can be analyzed as follows.

When T is a *q-parallel* DHBE tuple, $\mathcal{A}$ has an advantage ϵ by definition. We have $Pr[\nu = \nu' | T = e(g,g)^{a^{q+1}s_0}] = \frac{1}{2} + \epsilon$. Under the above policy, we have

$$Pr[\mathcal{B}(\vec{y}, e(g,g)^{a^{q+1}s_0}) = 0] = \frac{1}{2} + \epsilon. \tag{15}$$

With a random T, ν can be completely hidden with probability $\frac{1}{2}$ to successfully guess it. Under the definition of Eq. (1), $Adv_{\mathcal{B}} = \epsilon$.

Since $Adv_{\mathcal{B}}$ is non-negligible, which is contrary to the decisional *q-parallel* BDHE assumption, we conclude that our scheme is semantically secure. □

We additionally focus on the adversary who has relevant identical sub-policy parameter. This indicates that the adversary's attribute set S_j satisfies (M_s, ρ_s), but M' cannot be satisfied with S_j. The adversary may assert an attribute $S'_j \subset S_j$ that satisfies M', and a set of constants $\{\omega_i\}$ in output. Turn to Eq. (9), the adversary can compute:

$$\mathcal{K} / \left(\prod_{i \in I, \rho'(i) \in S'_j - S_j} e(C_i, L) e(C'_i, K_{\rho'(i)})^{\omega_i} \right),$$

where $K_{\rho'(i)} \in S'_j - S_j$ is the attribute-related key, and $\rho(i)$ is the attribute asserted by the adversary, but is not in his/her actual attribute set. To forge a random $K_{\rho'(i)} \in \mathbb{G}_1$, the final $\mathcal{K}^*$ will be any element in $\mathbb{G}_2$, and the probability of $\mathcal{K}^* = \mathcal{K}$ equals to q^{-1}, which is negligible.

7.3 Performance Analysis

This section evaluates the performance of our proposed scheme in terms of computation, communication, and storage costs, compared with Waters' Scheme [23], which is widely used in some related schemes, such as [9], [37], [38]. It should be noted that based on the traditional CP-ABE, numerous schemes have been proposed from different

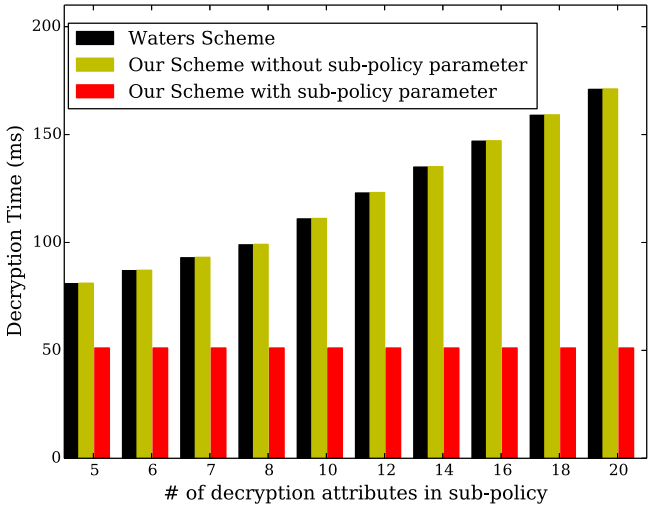


Fig. 4. Comparison of decryption time.

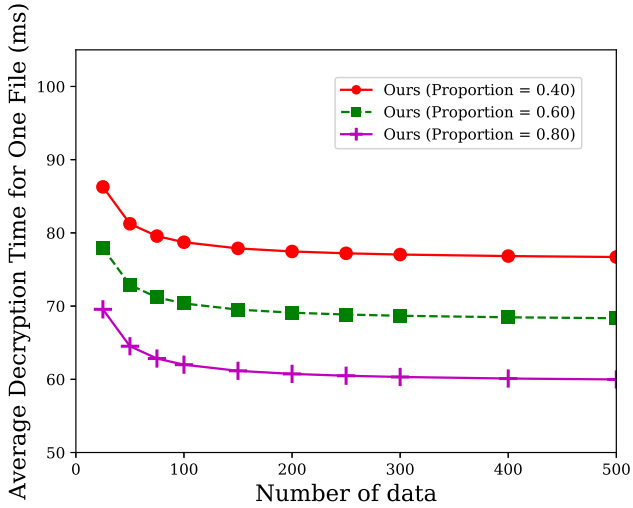


Fig. 6. Average decryption cost versus number of data.

perspectives to achieve performance improvements and make them more practical. These innovative solutions include online/offline encryption (e.g., [37], [38]), encryption/decryption outsource (e.g., [9], [11]), collaborative access control (e.g., [13]), attribute revocation (e.g., [52]) and so on. Our scheme is also improved on the traditional CP-ABE to achieve performance optimization when there're frequently used sub-policies among different files. The above mentioned schemes and ours aim at different optimization goals and are suitable for different application scenarios. But these schemes can also be integrated with our scheme to exert respective advantages. Therefore, we only choose the original representative scheme, i.e., Waters' scheme, as the compared one and conduct the performance comparison.

7.3.1 Computation Cost

We evaluate the computation time of decryption coded in a C program with PBC library 0.5.14 with type-A curve. The experiment is conducted in a standard 64-bit Fedora release 21 operation system with Inter(R) Core(TM) i3-4130 3.40 GHz.

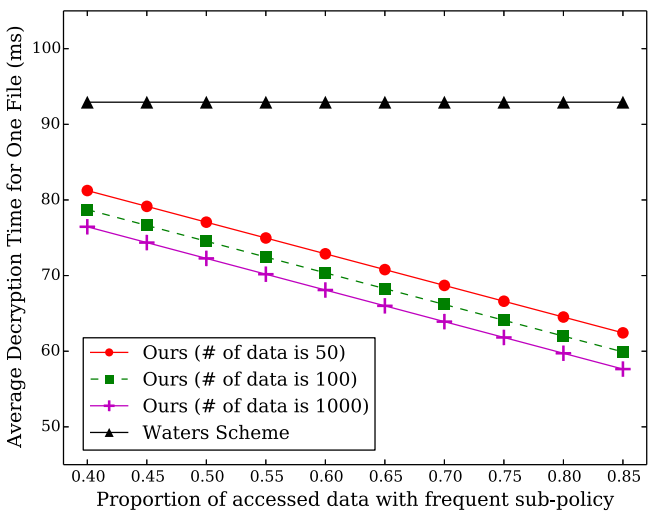


Fig. 5. Average decryption cost versus proportion of accessed data with frequent sub-policy.

For a data owner, the encryption time are around the same between Waters Scheme and our proposed scheme, no matter whether the comparison is with the encryption with identical sub-policy parameters or not. This is because the cost of random vector generation is negligible.

For users, the decryption time for one file is shown in Fig. 4 with respect to the complexity of sub-policy. From the figure, a user without a parameter takes almost the same time to decrypt though our scheme executes one more operation of exponentiation. However, the computation cost of the user's future decryption can be largely reduced, as there is no need to re-decrypt the portion identified by the parameters.

With this result, we also simulate a scenario to evaluate user's average decryption time with respect to the proportion of data embedded with frequently used sub-policy. Figs. 5 and 6 shows the evaluation result from two different perspectives, respectively. In this simulation, we consider the scenario with 6 different sub-policies. For each sub-policy, we simulate the first decryption without parameter, and some subsequent decryptions with relevant parameter. The proportion of data with frequently used sub-policies varies from 40 to 85 percent. Based on this measurement, the conclusions on our research are as follows: 1) Our scheme can reduce more decryption cost when there are larger proportion of data with frequently used sub-policies; 2) As the scale of accessed data increases, the greater the computation cost saved will be. However, the decreasing of the average decryption time for one file slows down with the increase of the accessed data scale, and gets closer and closer to the cost for a file when using sub-policy.

7.3.2 Storage and Communication Cost

Suppose that the system uses 2 bytes for a sequence ID (seq_s), and $|p|$ is the element size of $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{Z}_p$ (about 625 bytes in our experiment). Table 3 compares the storage and communication costs.

In our scheme, a user pays additional storage cost to maintain identical sub-policy parameters, whereas, the size of the parameter for a sub-policy is extremely small compared with the secret key SK, as the number of stored sub-policies is much smaller than user's attribute number, and

TABLE 3
Storage and Communication Cost

	Waters Scheme	Our Scheme
Storage		
User	$(2+n)\|p\|$	$(2+n)\|p\|$ $+(v_{M_S}+2+2\|p\|)n_{cs}$
Owner	N/A	$v_{M_S}+2+n_a\|p\|$
Communication		
Data	$(2+2n)\|p\|$ $+\|\mathcal{M}\|+v_M$	$(2+2n)\|p\|+\|\mathcal{M}\|$ $+v_M+v_{M_S}+2$

n: number of attributes assigned to the user;
n_{cs}: number of sub-policies cached by the user;
n_a: number of attributes in the sub-policy;
v_M/v_{M_S}: volume of description for access policy and sub-policy, respectively.
$\mathcal{M}$: volume of $E_{\mathcal{K}}(\mathcal{M})$;
$|p|$: element size of $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{Z}_p$.

the volume of sub-policy $v_{M_S} \ll |p|$. The storage of the owner does not take the shared data and system's public parameter into account. Thus, in our comparison, the existing schemes need no storage on the owners' side, while in our scheme, the owner should securely store the sub-policy. However, in practical scenarios, this burden is affordable for the owner.

The communication cost is mainly for the ciphertext to be uploaded and downloaded. For each data, our scheme takes a little more burden because of the existence of (M_s, ρ_s) and seq_s, which is negligible compared with other components in the data.

From the performance analysis, our scheme shows its significant advantage on computation time reduction for the data with frequently used sub-policies. Meanwhile, our scheme trades in only very little storage overhead.

8 Conclusion

In this paper, we presented an efficient and secure attribute-based access control scheme for the scenarios where user's accessed data are embedded with frequently used sub-policies. With the proposed mechanism of using identical sub-policy parameters, our scheme removes the repeated and redundant computation burden for the decryptions of different files with identical sub-policy. More specifically, in our design, the decryption process for the first data access assists the decryptions of subsequent relevant data with identical sub-policies in their access policies.

Besides, to leverage the decryption computation, the owner and user just need very small storage to maintain their parameters. The analysis also witnessed the significant improvements in decryption efficiency and security preservation of our proposed scheme. Our proposed scheme remarkably promotes the efficiency of access control for the scenarios where identical sub-policies are frequently embedded in sufficient shared data, and such scenarios usually appears in cloud storage.

Acknowledgments

The authors sincerely thank all the anonymous referees for their invaluable suggestions that have led to the present improved version from the original manuscript. This work was supported in part by Youth Innovation Promotion Association of the Chinese Academy of Sciences (CAS) under Grant No. 2016394 and the National Natural Science Foundation of China under Grant No. 61972371.

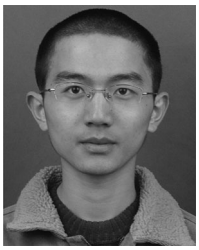
References

[1] M. Armbrust *et al.*, "A view of cloud computing," *Commun. ACM*, vol. 53, no. 4, pp. 50–58, 2010.
[2] L. Jiang, L. Da Xu, H. Cai, Z. Jiang, F. Bu, and B. Xu, "An IoT-oriented data storage framework in cloud computing platform," *IEEE Trans. Ind. Inform.*, vol. 10, no. 2, pp. 1443–1451, May 2014.
[3] H. Li, Y. Yang, T. H. Luan, X. Liang, L. Zhou, and X. Shen, "Enabling fine-grained multi-keyword search supporting classified sub-dictionaries over encrypted cloud data," *IEEE Trans. Dependable Secure Comput.*, vol. 13, no. 3, pp. 312–325, Jun. 2016.
[4] H. Li, D. Liu, Y. Dai, T. H. Luan, and S. Yu, "Personalized search over encrypted data with efficient and secure updates in mobile clouds," *IEEE Trans. Emerg. Topics Comput.*, vol. 6, no. 1, pp. 97–109, First Quarter 2018.
[5] X. Ma, Y. Zhao, L. Zhang, H. Wang, and L. Peng, "When mobile terminals meet the cloud: Computation offloading as the bridge," *IEEE Netw.*, vol. 27, no. 5, pp. 28–33, Sep./Oct. 2013.
[6] T. Hobfeld, R. Schatz, M. Varela, and C. Timmerer, "Challenges of QoE management for cloud applications," *IEEE Commun. Magazine*, vol. 50, no. 4, pp. 28–36, Apr. 2012.
[7] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *Proc. 28th IEEE Symp. Security Privacy*, 2007, pp. 321–334.
[8] Z. Wan, J. Liu, and R. H. Deng, "HASBE: A hierarchical attribute-based solution for flexible and scalable access control in cloud computing," *IEEE Trans. Inf. Forensics Security*, vol. 7, no. 2, pp. 743–754, Apr. 2012.
[9] K. Yang, X. Jia, K. Ren, B. Zhang, and R. Xie, "DAC-MACS: Effective data access control for multi-authority cloud storage systems," *IEEE Trans. Inf. Forensics Security*, vol. 8, no. 11, pp. 1790–1801, 2013.
[10] M. Li, S. Yu, Y. Zheng, K. Ren, and W. Lou, "Scalable and secure sharing of personal health records in cloud computing using attribute-based encryption," *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 1, pp. 131–143, Jan. 2013.
[11] H. Ma, R. Zhang, Z. Wan, Y. Lu, and S. Lin, "Verifiable and exculpable outsourced attribute-based encryption for access control in cloud computing," *IEEE Trans. Dependable Secure Comput.*, vol. 14, no. 6, pp. 679–692, Dec. 2017.
[12] K. Xue, W. Chen, W. Li, J. Hong, and P. Hong, "Combining data owner-side and cloud-side access control for encrypted cloud storage," *IEEE Trans. Inf. Forensics Security*, vol. 13, no. 8, pp. 2062–2074, Aug. 2018.
[13] Y. Xue, K. Xue, N. Gai, J. Hong, D. S. Wei, and P. Hong, "An attribute-based controlled collaborative access control scheme for public cloud storage," *IEEE Trans. Inf. Forensics Security*, vol. 14, no. 11, pp. 2927–2942, Nov. 2019.
[14] K. Ren, C. Wang, and Q. Wang, "Security challenges for the public cloud," *IEEE Internet Comput.*, vol. 16, no. 1, pp. 69–73, Jan./Feb. 2012.
[15] R. Masood, M. A. Shibli, Y. Ghazi, A. Kanwal, and A. Ali, "Cloud authorization: Exploring techniques and approach towards effective access control framework," *Frontiers Comput. Sci.*, vol. 9, no. 2, pp. 297–321, 2015.
[16] H. Li, Y. Yang, Y. Dai, J. Bai, S. Yu, and Y. Xiang, "Achieving secure and efficient dynamic searchable symmetric encryption over medical cloud data," *IEEE Trans. Cloud Comput.*, to be published, doi: 10.1109/TCC.2017.2769645.
[17] G. Xu, H. Li, Y. Dai, K. Yang, and X. Lin, "Enabling efficient and geometric range query with access control over encrypted spatial data," *IEEE Trans. Inf. Forensics Security*, vol. 14, no. 4, pp. 870–885, Apr. 2019.
[18] J. Shao, R. Lu, and X. Lin, "Fine-grained data sharing in cloud computing for mobile devices," in *Proc. 34th IEEE Int. Conf. Comput. Commun.*, 2015, pp. 2677–2685.
[19] J. Hong *et al.*, "TAFC: Time and attribute factors combined access control on time-sensitive data in public cloud," *IEEE Trans. Services Comput.*, vol. 13, no. 1, pp. 158–171, Jan./Feb. 2020.
[20] K. Xue *et al.*, "RAAC: Robust and auditable access control with multiple attribute authorities for public cloud storage," *IEEE Trans. Inf. Forensics Security*, vol. 12, no. 4, pp. 953–967, Apr. 2017.
[21] J. Hong, K. Xue, N. Gai, D. Wei, and P. Hong, "Service outsourcing in F2C architecture with attribute-based anonymous access control and bounded service number," *IEEE Trans. Dependable Secure Comput.*, to be published, doi: 10.1109/TDSC.2018.2845381.

[22] Y. Zhu, D. Huang, C.-J. Hu, and X. Wang, "From RBAC to ABAC: Constructing flexible data access control for cloud storage services," *IEEE Trans. Services Comput.*, vol. 8, no. 4, pp. 601–616, Jul./Aug. 2015.
[23] B. Waters, "Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization," in *Proc. 14th Int. Conf. Practice Theory Public Key Cryptography*, 2011, pp. 53–70.
[24] A. Lewko and B. Waters, "Decentralizing attribute-based encryption," in *Proc. 30th Annu. Int. Conf. Theory Appl. Cryptographic Techn.*, 2011, pp. 568–588.
[25] P.-W. Chi and C.-L. Lei, "Audit-free cloud storage via deniable attribute-based encryption," *IEEE Trans. Cloud Comput.*, vol. 6, no. 2, pp. 414–427, Apr.–Jun. 2018.
[26] S. Ruj, A. Nayak, and I. Stojmenovic, "DACC: Distributed access control in clouds," in *Proc. 10th IEEE Int. Conf. Trust Secur. Privacy Comput. Commun.*, 2011, pp. 91–98.
[27] J. Han, W. Susilo, Y. Mu, J. Zhou, and M. H. A. Au, "Improving privacy and security in decentralized ciphertext-policy attribute-based encryption," *IEEE Trans. Inf. Forensics Security*, vol. 10, no. 3, pp. 665–678, Mar. 2015.
[28] W. Li, K. Xue, Y. Xue, and J. Hong, "TMACS: A robust and verifiable threshold multi-authority access control system in public cloud storage," *IEEE Trans. Parallel Distrib. Syst.*, vol. 27, no. 5, pp. 1484–1496, May 2016.
[29] Z. Liu, Z. Cao, and D. S. Wong, "White-box traceable ciphertext-policy attribute-based encryption supporting any monotone access structures," *IEEE Trans. Inf. Forensics Security*, vol. 8, no. 1, pp. 76–88, Jan. 2013.
[30] J. Zhou, Z. Cao, X. Dong, and X. Lin, "TR-MABE: White-box traceable and revocable multi-authority attribute-based encryption and its applications to multi-level privacy-preserving e-healthcare cloud computing systems," in *Proc. 34th IEEE Int. Conf. Comput. Commun.*, 2015, pp. 2398–2406.
[31] K. Yang, X. Jia, K. Ren, R. Xie, and L. Huang, "Enabling efficient access control with dynamic policy updating for big data in the cloud," in *Proc. 33rd IEEE Int. Conf. Comput. Commun.*, 2014, pp. 2013–2021.
[32] S. Yu, C. Wang, K. Ren, and W. Lou, "Achieving secure, scalable, and fine-grained data access control in cloud computing," in *Proc. 29th IEEE Int. Conf. Comput. Commun.*, 2010, pp. 1–9.
[33] K. Yang, X. Jia, and K. Ren, "Attribute-based fine-grained access control with efficient revocation in cloud storage systems," in *Proc. 8th ACM SIGSAC Symp. Inf. Comput. Commun. Secur.*, 2013, pp. 523–528.
[34] M. Li, W. Lou, and K. Ren, "Data security and privacy in wireless body area networks," *IEEE Wireless Commun.*, vol. 17, no. 1, pp. 51–58, Feb. 2010.
[35] 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.
[36] M. Green, S. Hohenberger, and B. Waters, "Outsourcing the decryption of ABE ciphertexts," in *Proc. 20th USENIX Secur. Symp.*, 2011, p. 3.
[37] S. Hohenberger and B. Waters, "Online/offline attribute-based encryption," in *Proc. 17th Int. Conf. Practice Theory Public Key Cryptography*, 2014, pp. 293–310.
[38] Y. Zhang, D. Zheng, Q. Li, J. Li, and H. Li, "Online/offline unbounded multi-authority attribute-based encryption for data sharing in mobile cloud computing," *Secur. Commun. Netw.*, vol. 9, no. 16, pp. 3688–3702, 2016.
[39] J. Ning, Z. Cao, X. Dong, K. Liang, H. Ma, and L. Wei, "Auditable σ-time outsourced attribute-based encryption for access control in cloud computing," *IEEE Trans. Inf. Forensics Secur.*, vol. 13, no. 1, pp. 94–105, Jan. 2018.
[40] V. Kolesnikov, H. Krawczyk, Y. Lindell, A. J. Malozemoff, and T. Rabin, "Multi-cloud oblivious storage," in *Proc. 23th ACM Conf. Comput. Commun. Secur.*, 2016, pp. 247–258.
[41] S. Hohenberger and B. Waters, "Attribute-based encryption with fast decryption," in *Proc. 16th Int. Conf. Practice Theory Public Key Cryptography*, 2013, pp. 162–179.
[42] T. Zhao, L. Wei, and C. Zhang, "Attribute-based encryption scheme based on SIFF," in *Proc. IEEE Int. Conf. Commun.*, 2016, pp. 1–6.
[43] S. Wang, J. Zhou, J. K. Liu, J. Yu, J. Chen, and W. Xie, "An efficient file hierarchy attribute-based encryption scheme in cloud computing," *IEEE Trans. Inf. Forensics Security*, vol. 11, no. 6, pp. 1265–1277, Jun. 2016.
[44] G. Wang, Q. Liu, and J. Wu, "Hierarchical attribute-based encryption for fine-grained access control in cloud storage services," in *Proc. 17th ACM Conf. Comput. Commun. Secur.*, 2010, pp. 735–737.
[45] D. Boneh and M. Franklin, "Identity-based encryption from the Weil pairing," in *Proc. 21th Annu. Int. Cryptol. Conf.*, 2001, pp. 213–229.
[46] D. Boneh, B. Lynn, and H. Shacham, "Short signatures from the weil pairing," in *Proc. Int. Conf. Theory Appl. Cryptol. Inf. Secur.*, 2001, pp. 514–532.
[47] 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.*, 2006, pp. 89–98.
[48] Z. Liu, Z. Cao, and D. S. Wong, "Efficient generation of linear secret sharing scheme matrices from threshold access trees," *IACR Cryptol. ePrint Archive*, vol. 2010, 2010, Art. no. 374, [Online]. Available: https://eprint.iacr.org/2010/374.pdf
[49] L. Cheung and C. Newport, "Provably secure ciphertext policy ABE," in *Proc. 14th ACM Conf. Comput. Commun. Secur.*, 2007, pp. 456–465.
[50] J. Herranz, F. Laguillaumie, and C. Ràfols, "Constant size ciphertexts in threshold attribute-based encryption," in *Proc. 13th Int. Conf. Practice Theory Public Key Cryptography*, pp. 19–34, 2010.
[51] C. Zuo, J. Shao, J. K. Liu, G. Wei, and Y. Ling, "Fine-grained two-factor protection mechanism for data sharing in cloud storage," *IEEE Trans. Inf. Forensics Security*, vol. 13, no. 1, pp. 186–196, Jan. 2018.
[52] J. K. Liu, T. H. Yuen, P. Zhang, and K. Liang, "Time-based direct revocable ciphertext-policy attribute-based encryption with short revocation list," in *Proc. Int. Conf. Appl. Cryptography Netw. Secur.*, 2018, pp. 516–534.

Kaiping Xue (Senior Member, IEEE) received the bachelor's degree from the Department of Information Security, University of Science and Technology of China (USTC), in 2003, and the PhD degree from the Department of Electronic Engineering and Information Science (EEIS), USTC, in 2007. From May 2012 to May 2013, he was a postdoctoral researcher with the Department of Electrical and Computer Engineering, University of Florida. Currently, he is an associate professor with the School of Cyber Security and the Department of EEIS, USTC. His research interests include next-generation Internet, distributed networks, and network security. He has authored and co-authored more than 80 technical papers in the areas of communication networks and network security. His work won Best Paper Awards in IEEE MSN 2017, IEEE HotICN 2019, and Best Paper Runner-Up Award in IEEE MASS 2018. He serves on the editorial board of several journals, including the *IEEE Transactions on Wireless Communications* (TWC), *IEEE Transactions on Network and Service Management* (TNSM), Ad Hoc Networks, IEEE Access and China Communications. He has also served as a guest editor of the *IEEE Journal on Selected Areas in Communications* (JSAC) and a lead guest editor of the *IEEE Communications Magazine*. He is serving as the program co-chair for IEEE IWCMC 2020 and SIGSAC@TURC 2020. He is an IET fellow.

Na Gai received the BS degree from the Department of Information Security, University of Science and Technology of China (USTC), in 2018. She is currently working toward the graduated degree in information security from the Department of Electronic Engineering and Information Science (EEIS), USTC. Her research interests include network security protocol design and analysis.



Jianan Hong received the BS degree from the Department of Information Security, University of Science and Technology of China (USTC), in 2012, and the PhD degree from the Department of Electronic Engineering and Information Science (EEIS), USTC, in 2018. Currently he is a research Engineer in Huawei Shanghai Research Institute, Shanghai. His research interests include secure cloud computing and mobile network security.

David S.L. Wei (Senior Member, IEEE) received the PhD degree in computer and information science from the University of Pennsylvania, in 1991. He is currently a full professor of Computer and Information Science Department, Fordham University. From May 1993 to August 1997 he was in the Faculty of Computer Science and Engineering, University of Aizu, Japan (as an associate professor and then a full professor). He has authored and co-authored more than 120 technical papers in the areas of distributed and parallel processing, wireless networks and mobile computing, optical networks, peer-to-peer communications, cognitive radio networks, big data, cloud computing, and IoT in various archival journals and conference proceedings. He served on the program committee and was a session chair for several reputed international conferences. He was a lead guest editor of *IEEE Journal on Selected Areas in Communications for the special issue on Mobile Computing and Networking*, a lead guest editor of *IEEE Journal on Selected Areas in Communications for the special issue on Networking Challenges in Cloud Computing Systems and Applications*, a guest editor of *IEEE Journal on Selected Areas in Communications for the special issue on Peer-to-Peer Communications and Applications*, a lead guest editor of *IEEE Transactions on Cloud Computing* for the special issue on Cloud Security, a guest editor of *IEEE Transactions on Big Data for the special issue on Trustworthiness in Big Data and Cloud Computing Systems*, and a lead guest editor of *IEEE Transactions on Big Data for the special issue on Edge Analytics in the Internet of Things.* He also served as an associate editor of *IEEE Transactions on Cloud Computing*, 2014–2018, and an associate editor of *Journal of Circuits, Systems and Computers*, 2013–2018. He is presently an editor of *IEEE Journal on Selected Areas in Communications for the Series on Network Softwarization & Enablers* and a lead guest editor of *IEEE Journal on Selected Areas in Communications for the special issue on Leveraging Machine Learning in SDN/NFV-based Networks*. Currently, He focuses his research efforts on cloud and edge computing, IoT, big data, machine learning, and cognitive radio networks.

Peilin Hong received the BS and MS degrees from the Department of Electronic Engineering and Information Science (EEIS), University of Science and Technology of China (USTC), in 1983 and 1986. Currently, she is a professor and advisor for PhD candidates with the Department of EEIS, USTC. Her research interests include next-generation Internet, policy control, IP QoS, and information security. She has published two books and more than 150 academic papers in several journals and conference proceedings.

Nenghai Yu received the BS degree from the Nanjing University of Posts and Telecommunications, Nanjing, China, in 1987, the ME degree from Tsinghua University, Beijing, China, in 1992, and the PhD degree from the Department of Electronic Engineering and Information Science (EEIS), University of Science and Technology of China (USTC), Hefei, China, in 2004. Since 1992, he has been a Faculty in the Department of Electronic Engineering and Information Science, USTC, where he is currently a Professor. He is the executive director of the Department of EEIS, USTC, and the director of the Information Processing Center, USTC. He has authored or co-authored more than 130 papers in journals and international conferences. His research interests include multimedia security, multimedia information retrieval, video processing, and information hiding.