Fine-Grained Access Control System Based on Outsourced Attribute-Based Encryption

Jin Li[1], Xiaofeng Chen[2], Jingwei Li[3], Chunfu Jia[3], Jianfeng Ma[4], and Wenjing Lou[5]

[1] School of Computer Science and Educational Software, Guangzhou University, P.R. China
jinli71@gmail.com

[2] State Key Laboratory of Integrated Service Networks, Xidian University, P.R. China
xfchen@xidian.edu.cn

[3] College of Information Technical Science, Nankai University, P.R. China
lijw@mail.nankai.edu.cn, cfjia@nankai.edu.cn

[4] School of Computer Science and Technology, Xidian University, P.R. China
jfma@mail.xidian.edu.cn

[5] Department of Computer Science, Virginia Polytechnic Institute and State University, USA
wjlou@vt.edu

Abstract. As cloud computing becomes prevalent, more and more sensitive data is being centralized into the cloud for sharing, which brings forth new challenges for outsourced data security and privacy. Attribute-based encryption (ABE) is a promising cryptographic primitive, which has been widely applied to design fine-grained access control system recently. However, ABE is being criticized for its high scheme overhead as the computational cost grows with the complexity of the access formula. This disadvantage becomes more serious for mobile devices because they have constrained computing resources.

Aiming at tackling the challenge above, we present a generic and efficient solution to implement attribute-based access control system by introducing secure outsourcing techniques into ABE. More precisely, two cloud service providers (CSPs), namely key generation-cloud service provider (KG-CSP) and decryption-cloud service provider (D-CSP) are introduced to perform the outsourced key-issuing and decryption on behalf of attribute authority and users respectively. In order to outsource heavy computation to both CSPs without private information leakage, we formulize an underlying primitive called outsourced ABE (OABE) and propose several constructions with outsourced decryption and key-issuing. Finally, extensive experiment demonstrates that with the help of KG-CSP and D-CSP, efficient key-issuing and decryption are achieved in our constructions.

1 Introduction

Cloud computing is an emerging computing paradigm in which IT resources and capacities are provided as services over the Internet while hiding platform and implementation details. Promising as it is, this paradigm also brings forth new challenges for data security and privacy when users outsource sensitive data for sharing on cloud servers, which are likely outside of the same trusted domain of data owners.

Data access control has been evolving in the past thirty years and various techniques have been developed to effectively implement fine-grained access control [20], which allows flexibility in specifying differential access rights of individual users. However, traditional access control systems are mostly designed for in-house services and depend greatly on the system itself to enforce authorization policies. Thus, they cannot be applied in cloud computing because users and cloud servers are no longer in the same trusted domain. For the purpose of helping the data owner impose access control over data stored on untrusted cloud servers, a feasible consideration would be encrypting data through certain cryptographic primitives but disclosing decryption keys only to authorized users. One critical issue of this branch of approaches is how to achieve the desired security goals without introducing high complexity of key management and data encryption. Existing work resolve this issue either by introducing a per file access control list (ACL) for fine-grained access control, or by categorizing files into several filegroups for efficiency. As the system scales, however, the ACL-based scheme would introduce an extremely high complexity which could be proportional to the number of system users. The filegroup-based scheme, on the other hand, is just able to provide coarse-grained access control of data.

Aiming at providing fine-grained access control over encrypted data, a novel public key primitive namely attribute-based encryption (ABE) [23] is introduced in the cryptographic community, which enables public key-based one-to-many encryption. In ABE system, users' keys and ciphertexts are labeled with sets of descriptive attributes and access policies respectively, and a particular key can decrypt a ciphertext only if the associated attributes and policy are matched.

Though ABE is a promising primitive to design fine-grained access control system in cloud computing, there are several challenges remained in the application of ABE.

- One of the main drawbacks of ABE is that the computational cost in decryption phase grows with the number of attributes specified in the access policy. The drawback appears more serious for resource-constrained users such as mobile devices and sensors. Therefore, one challenge is *how to reduce the decryption complexity of ABE such that it can be applied to fine-grained access control for users with resource-constrained devices.*
- Beyond decryption, generating user's private key in existing ABE schemes also requires a great quantity of modular exponentiations. Furthermore, the revocation of any single user in existing ABE requires key-update at authority for remaining users who share his/her attributes. All of these heavy tasks

centralized at authority side would make it become the efficiency bottleneck in the whole access control system. Therefore, another challenge is *how to reduce the key-issuing complexity of ABE such that scalable access control can be supported.*

1.1 Contribution Overview

Aiming at tackling the challenges described above, we propose a generic construction of attribute-based access control system under an interesting architecture, in which two cloud service providers (CSPs) namely key generation-cloud service provider (KG-CSP) and decryption-cloud service provider (D-CSP) are involved to perform the outsourced heavy tasks for users' key issuing and file access. With the help of the CSPs, the computational complexity at both user and attribute authority sides is reduced. Furthermore, since only small computation is required at authority side for single user's private key update, the proposed system is able to efficiently support user revocation.

The challenge issue in the proposed system is how to outsource the heavy computation to the CSPs as much as possible but without private information leakage. Our solution is introducing an underlying primitive namely outsourced ABE (OABE), which allows expensive tasks to be securely outsourced to CSPs to relieve computation overhead at local. We provide several OABE constructions with outsourced key-issuing and decryption. As far as we know, this work is the first attempt considering outsourcing key-issuing and decryption in ABE simultaneously. Our first construction requires only constant computation (nearly two single-based exponentiations) at attribute authority during key-issuing besides efficient decryption. Our second construction provides access control in a fine-grained manner but remains the same efficiency as previous constructions.

1.2 Organization

The rest of this paper is organized as follows. In Section 2, we present the architecture and adversary model for attribute-based access control system. In Section 3, an efficient access control system based on OABE is described. In Section 4, we propose a basic OABE construction with outsourced decryption for access control. Several OABE constructions with outsourced key-issuing and decryption for improved access control are presented in Section 5. In Section 6, an extensive experimental result is provided for demonstrating the efficiency of our main OABE construction. In Section 7, the previous work related to ours is surveyed. Finally, we draw conclusion in Section 8.

2 Attribute-Based Access Control System Model

In this section, we describe the architecture for the attribute-based access control system and define its security model.

2.1 Introduction of Attribute-Based Encryption

ABE has been widely applied to impose fine-grained access control on encrypted data recently. There are two kinds of ABE having been proposed: key-policy attribute-based encryption (KP-ABE) and ciphertext-policy attribute-based encryption (CP-ABE). In KP-ABE, each ciphertext is labeled by the encryptor with a set of descriptive attributes. Each private key is associated with an access structure that specifies which type of ciphertexts the key can decrypt. Whereas, in CP-ABE, the access structure is specified in ciphertext by encryptor and each private key is associated with a set of attributes. Without loss of generality, we are able to denote $(I_{\mathrm{enc}}, I_{\mathrm{key}})$ as the input to encryption and key generation of ABE. Accordingly, in CP-ABE scheme, $(I_{\mathrm{enc}}, I_{\mathrm{key}}) = (\mathbb{A}, \omega)$ while that is $(\omega, \mathbb{A})$ in KP-ABE, where ω and $\mathbb{A}$ denotes an attribute set and an access structure, respectively. In ABE, there are two entities: the attribute authority and users. The attribute authority is in charge of the issue of attribute private key to users requesting them. In more detail, the definition of four algorithms in ABE is given as follows.

– $\mathsf{Setup}(\lambda)$: The setup algorithm takes as input – a security parameter λ. It outputs (PK, MK), where PK denotes the public key and MK denotes the master key of the attribute authority.
– $\mathsf{KeyGen}(I_{\mathrm{key}}, MK)$: The key extraction algorithm takes as input – a user's access structure (resp. attribute set) I_{key} and the attribute authority's master key MK. It outputs the user's private key SK.
– $\mathsf{Encrypt}(M, I_{\mathrm{enc}})$: The encryption algorithm takes as input – a message M and the attribute set (resp. access structure) I_{enc}. It outputs the ciphertext CT with access policy I_{enc}.
– $\mathsf{Decrypt}(CT, SK)$: The decryption algorithm takes as input – a ciphertext CT which was assumed to be encrypted under the attribute set (resp. access structure) I_{enc} and the private key SK for access structure (resp. attribute set) I_{key}. It outputs the message M if $\gamma(I_{\mathrm{key}}, I_{\mathrm{enc}}) = 1$ and the error symbol $\perp$ otherwise, where the predicate γ is predefined.

2.2 Architecture for the Attribute-Based Access Control System

As shown in Fig. 1, the architecture for the attribute-based access control system consists of the following entities:

– *Attribute Authority (AA).* This is a key authority for the attribute set. It is in charge of issuing, revoking, and updating attribute keys for users.
– *Data Owner.* This is a user who owns data files and wishes to outsource them into the external storage server provided by a CSP. It is responsible for defining and enforcing an attribute set (resp. access policy) on its own files.
– *User.* This is an entity who wants to access an outsourced file. If the user owns an access privilege of an encrypted file, and is not revoked, he/she will be able to obtain the file.

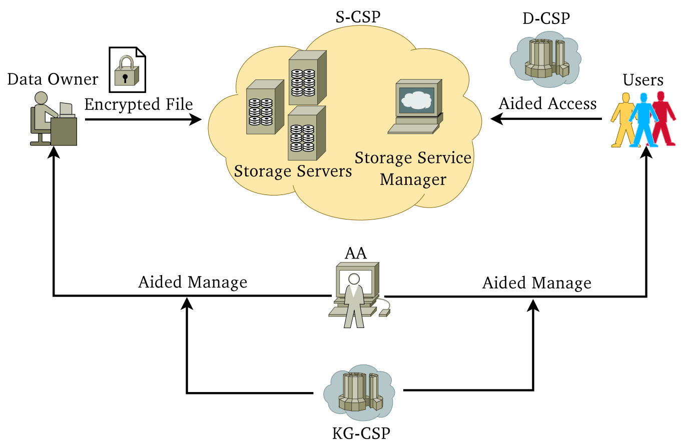


Fig. 1. Architecture for Attribute-based Access Control System

- *Storage-Cloud Service Provider (S-CSP).* This is an entity that provides a data storage outsourcing service. In this paper, we assume that S-CSP is always online and has abundant storage capacity and computation power.
- *Key Generation-Cloud Service Provider (KG-CSP).* This is an entity that provides an outsourcing computing service for AA through undertaking the expensive tasks delegated by AA.
- *Decryption-Cloud Service Provider (D-CSP).* This is an entity that provides an outsourcing computing service through performing partial decryption on ciphertext.

We give an overview of the attribute-based access control system as follows.

- **System Setup.** Public parameter and master key are initialized for the system and AA keeps the master key as secret information.
- **New User Grant.** When a new user wants to join the system, with the aid of KG-CSP, AA issues an attribute private key to him/her based on his/her attributes .
- **New File Creation.** When a data owner wants to outsource and share a file with some users, he/she encrypts the file to be uploaded under a specified attribute set (resp. access policy).
- **File Access.** When a user wants to access an outsourced file, he/she downloads ciphertext from S-CSP and decrypts it with the help of D-CSP.
- **User Revocation.** When there is a user to be revoked, AA updates "affected" users' private keys with the help of KG-CSP, while the "affected" ciphertexts having been stored on S-CSP will be updated as well.

2.3 Adversary Model and Security Requirements

We assume that S-CSP, D-CSP and KG-CSP are honest but curious. More precisely, they will follow our proposed protocols, but try to find out as much secret information as possible based on their possessions. Furthermore, D-CSP is allowed to collude with curious users and S-CSP. Thus, two types of adversaries are considered in our access control system: i) users colluding with D-CSP and S-CSP; ii) semi-trusted KG-CSP, which is not allowed to collude with users.

The security requirement considered in this paper is *semantic security of data*, which is defined as follows: Unauthorized users (that is, the two types of adversaries defined above) without appropriate access structure (resp. attributes) matching the attributes (resp. policy) embedded in ciphertext should be prevented from accessing the underlying plaintext.

3 OABE-Based Access Control System

In this section, we provide a generic construction of the attribute-based access control system. Its security analysis is presented as well.

3.1 Building Block: OABE

Based on the system model provided in Section 2, we attempt to define an underlying primitive namely OABE with outsourced key-issuing and decryption for realizing our access control system. Notice that the definitions of Setup and Encrypt in OABE are identical to traditional ABE shown in Section 2.1, we only show the definitions of *outsourced key-issuing protocol* and *outsourced decryption protocol.*

Outsourced Key-Issuing Protocol. Three entities including users, AA and KG-CSP are involved in this protocol. Upon receiving a key-issuing request from a user, AA firstly sends an outsourcing key (denoted as OK) to KG-CSP and receives a private key component (denoted as SK_1) for the user. The other component SK_2 is computed locally by AA. At a high level, the protocol is described as follows.

$$\begin{tikzcd}\text{User} \arrow[r, shift left, "\text{request}"] & \text{AA} \arrow[l, shift left, "SK"] \arrow[r, shift left, "OK"] & \text{KG-CSP} \arrow[l, shift left, "SK_1"]\end{tikzcd}$$

The outsourced key-issuing protocol consists of the following three polynomial-time algorithms.

– O-KeyGen-PreProc(I_{key}, MK) : The preprocessing algorithm run by AA takes as input – the access structure (resp. attribute set) I_{key} for a user, the master key MK. It outputs the key pair (OK, AK) where OK denotes the outsourcing key for KG-CSP and AK denotes the secret key for AA to compute the other component of private key.

- O-KeyGen-Outsource(I_{key}, OK) : The outsourced algorithm run by KG-CSP takes as input – the access structure (resp. attribute set) I_{key} and the outsourcing key OK. It outputs the private key component SK_1.
- O-KeyGen-PostProc(AK, SK_1) : The postprocessing algorithm run by AA takes as input – the secret key AK and the private key SK_1. It outputs $SK = (SK_1, SK_2)$ as the user's private key.

Outsourced Decryption Protocol. Two entities including users and D-CSP are involved in this protocol. More precisely, upon receiving the ciphertext CT, the user delivers SK_1 along with CT to D-CSP and receives a partially decrypted ciphertext CT'. Finally, the message is completely computed by the user with SK. At a high level, it can be described as follows.

- O-Decrypt-Outsource(CT, SK_1) : The outsourced algorithm run by D-CSP takes as input – a ciphertext CT assumed to be encrypted under the attribute set (resp. access structure) I_{enc} and the private key component SK_1 for access structure (resp. attribute set) I_{key}. It outputs the partially decrypted ciphertext CT' if $\gamma(I_{\text{key}}, I_{\text{enc}}) = 1$, otherwise outputs $\perp$.
- O-Decrypt-Dec(CT', SK) : The complete decryption algorithm run by the user takes as input – the partially decrypted ciphertext CT' and the private key SK. It outputs a message M.

Security Model

Two types of adversaries are classified as in Section 2.3:

- *Type-I Adversary.* It is defined as a curious user colluding with D-CSP. Such an adversary is allowed to ask for all the SK_1 and the private keys SK of dishonest users. The goal of this adversary is to obtain useful information from ciphertext not intended for him/her. Notice that Type-I adversary cannot get outsourcing key OK for any user.
- *Type-II Adversary.* It is defined as a curious KG-CSP. Such an adversary owns outsourcing keys OK for all users in the system and tries to extract any useful information from ciphertext.

Having the intuition above, we are able to follow the replayable chosen-ciphertext attack (RCCA) security given in [15] and define it for both type-I and type-II adversaries in our OABE. The security definition is similar to the previous work [15], where the only difference is that an additional security game is defined to simulate the type-II adversary with the outsourcing keys of all users.

Definition 1 (RCCA Security). *An OABE scheme with outsourced key-issuing and decryption is secure against replayable chosen-ciphertext attack if all polynomial-time adversaries have at most a negligible advantage in the RCCA security games for both type-I and type-II adversaries.*

3.2 Generic Construction of OABE-Based Access Control System

System Setup. Choose a security parameter 1^λ and run the algorithm $\mathsf{Setup}(1^\lambda)$ of OABE to obtain the public parameter PK and the master key MK. The public parameter is then published, while the master key is kept by AA as a secret.

New File Creation. Whenever a data owner wants to create and upload a file $\mathcal{F}$ to S-CSP, he/she firstly defines an attribute set (resp. access structure) I_{enc} for this file. Then, the owner randomly picks a symmetric key K from the key space and encrypts the file $\mathcal{F}$ with K using standard symmetric key algorithm such as AES to obtain the ciphertext $CT_{\mathcal{F}}$. Later on, he/she runs the algorithm $\mathsf{Encrypt}(I_{\text{enc}}, K)$ of OABE to generate the ciphertext CT_K which is an encryption of the symmetric key with respect to I_{enc}. Finally, the data owner uploads the ciphertext $(CT_{\mathcal{F}}, CT_K)$ to S-CSP.

New User Grant. Assuming a user wants to join the system, he/she needs to be issued a private key on his/her access structure (resp. attribute set) I_{key} from AA who then runs the outsourced key-issuing protocol. In concrete, AA outsources the operation of key-issuing by running the algorithm of $\mathsf{O\text{-}KeyGen\text{-}PreProc}(I_{\text{key}}, MK)$ to obtain an outsourcing key OK. Using OK, KG-CSP runs $\mathsf{O\text{-}KeyGen\text{-}Outsource}(I_{\text{key}}, OK)$ to generate a private key component SK_1. Finally, AA generates the other private key component SK_2 and assigns $SK = (SK_1, SK_2)$ to the user.

File Access. Suppose a user wants to access and retrieve files of his/her interests. He/She firstly downloads the ciphertext $(CT_{\mathcal{F}}, CT_K)$. To decrypt the ciphertext while relieving the local computation overhead, the user runs the outsourced decryption protocol with D-CSP by sending CT_K and the private key component SK_1. If the user's I_{key} in SK_1 matches I_{enc} embedded in CT_K, D-CSP is able to successfully compute and return the partially decrypted ciphertext CT'_K. Upon receiving CT'_K, the user performs complete decryption to get the symmetric key K, with which he/she decrypts and retrieves the file $\mathcal{F}$.

User Revocation. Whenever there is a user to be revoked, a public parameter update technique in [24] is utilized. Specifically, AA determines a minimal set of attributes according to the user's I_{key} and updates the corresponding components in PK and MK. Then, AA updates private keys $SK = (SK'_1, SK_2)$ for all the "affected" users by running the outsourced key-issuing protocol with KG-CSP. Additionally, to update "affected" ciphertexts having been stored in S-CSP, a re-encrypting key is generated by AA to be sent to S-CSP. S-CSP uses such a key to update the "affected" ciphertexts with the latest version of PK. Notice that the main computation at AA side is updating private keys for "affected" users. Utilizing the outsourced key-issuing protocol, such complexity is minimized.

Security Analysis

Theorem 1. *The generic construction of access control system is semantically-secure if the underlying hybrid encryption satisfies RCCA security.*

Proof. The security will be analyzed based on the security model of access control system given in 2.3. More specifically, two types of adversaries will be considered here, that is, type-I and type-II adversaries. For each type adversary, we show how to construct a simulator to break the hybrid encryption at a high level.

Since the file is encrypted with a hybrid encryption as $(CT_{\mathcal{F}}, CT_K)$, to get any information about $\mathcal{F}$, the adversary should decrypt CT_K to retrieve the symmetric key K. However such a key is protected by OABE. Thus, in the above generic construction, a hybrid encryption has been utilized to encrypt the file. As we know, the above hybrid encryption scheme could achieve RCCA security if the following two conditions satisfy [9], that is, i) the OABE scheme is RCCA secure and ii) the symmetric key encryption scheme is CCA secure. Thus, data confidentiality can be reduced to the confidentiality security of the underlying OABE and symmetric key encryption [9]. Moreover, the privacy of OABE ciphertext on S-CSP against outside users without I_{key} can be trivially guaranteed because its security definition inherits that in traditional ABE. Another attack on data confidentiality is launched by KG-CSP. Such an attack is modeled as type-II adversary by introducing an oracle $\mathcal{O}_{OK}(\cdot)$ in the corresponding security game of OABE. Specifically, we allow such an adversary to ask for outsourcing keys OK for all the users, but it is not allowed to get any secret key of users. Curious users can collude with both D-CSP and S-CSP to launch attack, which is modeled as type-I adversary in the definition of OABE. Therefore, the security of the attribute-based access control system is reduced to that of underlying OABE and symmetric encryption.

In the following sections, we take our focus on OABE and attempt to provide secure OABE construction for attribute-based access control system.

4 Basic OABE Construction with Outsourced Decryption

4.1 Access Structure

Definition 2 (Access Structure). *Let $\{P_1, \dots, P_n\}$ be a set of parties. A collection $\mathbb{A} \subseteq 2^{\{P_1,\dots,P_n\}}$ is monotone if $\forall B, C$: if $B \in \mathbb{A}$ and $B \subseteq C$ then $C \in \mathbb{A}$. An access structure (resp. monotone access structure) is a collection (resp. monotone collection) $\mathbb{A}$ of non-empty subsets of $\{P_1, \dots, P_n\}$. The sets in $\mathbb{A}$ are called authorized sets.*

Denote ω and $\mathbb{A}$ as an attribute set and access structure, respectively. We define a predicate $\gamma(\omega, \mathbb{A})$ as follows

$$\gamma(\omega, \mathbb{A}) = \begin{cases} 1 \text{ if } \omega \in \mathbb{A} \\ 0 \text{ otherwise} \end{cases}$$

In this paper, the role of the party is taken by attributes. Thus, the access structure $\mathbb{A}$ contains the authorized sets of attributes. Specifically, the access structure represented by tree can be supported in this paper.

Let $\mathcal{T}$ be an access tree, in which each interior node is a threshold gate (i.e. AND gate or OR gate) while the leaves are associated with attributes. A user is able to decrypt a ciphertext with a given key if and only if there is an assignment of attributes from the private key to leaf nodes of the tree such that the tree is satisfied.

4.2 Basic Mathematical Tools

We introduce two basic mathematical tools that will be used in the following constructions.

Definition 3 (Bilinear Map). *Let $\mathbb{G}$ and $\mathbb{G}_T$ be cyclic groups of prime order q, writing the group action multiplicatively. g is a generator of $\mathbb{G}$. Let $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ be an efficient map with the following properties:*

- *Bilinearity: $e(g_1^a, g_2^b) = e(g_1, g_2)^{ab}$ for all $g_1, g_2 \in \mathbb{G}$, and $a, b \in_R \mathbb{Z}_q$;*
- *Non-degeneracy: There exists $g_1, g_2 \in \mathbb{G}$ such that $e(g_1, g_2) \neq 1$, in other words, the map does not send all pairs in $\mathbb{G} \times \mathbb{G}$ to the identity in $\mathbb{G}_T$.*

Definition 4 (DBDH assumption). *The decision Bilinear Diffie-Hellman (DBDH) assumption is that, given g, g^x, g^y, $g^z \in \mathbb{G}$ for unknown random values $x, y, z \in_R \mathbb{Z}_q$, and $T \in_R \mathbb{G}_T$, it is difficult to decide if $T = e(g, g)^{xyz}$ for a probabilistic polynomial algorithm.*

4.3 Proposed Construction

We only consider to outsource the decryption computation of ABE and propose a basic OABE construction with outsourced decryption. In another word, the KG-CSP will not be involved. For simplicity, this basic construction only considers to support for access structure described as $\mathbb{A} = \{\omega \subseteq \mathcal{U} : |\omega \cap \omega^*| \geq d\}$ where $\mathcal{U}$ is the attribute universe, ω and ω^* are attribute sets and d is a predefined threshold value. Actually, it can be easily extended to an OABE supporting access structure represented by tree as shown in Section 5.2.

Before providing the construction, we define the Lagrange coefficient $\Delta_{i,S}$ for $i \in \mathbb{Z}_p$ and a set S of elements in $\mathbb{Z}_p$ as $\Delta_{i,S} = \prod_{j \in S, j \neq i} \frac{x-j}{i-j}$.

Setup Phase

- $\mathsf{Setup}(\lambda)$: Define a bilinear group $\mathbb{G}$ of prime order p with a generator g and a bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. Next, define the attributes in universe $\mathcal{U}$ as elements in $\mathbb{Z}_p$. For simplicity let $n = |\mathcal{U}|$ and the first n elements in $\mathbb{Z}_p$ (i.e. $1, 2, \ldots, n \bmod p$) can be taken to be the universe. Select $x \in_R \mathbb{Z}_p$ and set $g_1 = g^x$. Pick $g_2, h, h_1, \ldots, h_n \in_R \mathbb{G}$. Output the public parameter $PK = (g, g_1, g_2, h, h_1, \ldots, h_n)$ and the master key $MK = x$ which is kept secret by AA.

Key-Issuing Phase. A hybrid policy $\mathcal{P} = \mathcal{P}_\theta \wedge \mathcal{P}_\omega$ is utilized in the key-issuing phase, where $\wedge$ is an AND gate connecting two sub-policies $\mathcal{P}_\omega$ and $\mathcal{P}_\theta$. More precisely, a default attribute θ is appended with each user's attribute set and the master key x is randomly split into x_1 and x_2 for each user to generate private key components on $\mathcal{P}_\omega$ and $\mathcal{P}_\theta$ respectively.

The key generation algorithm is described as follows.

– $\mathsf{KeyGen}(\omega, MK)$: Upon receiving a private key request on attribute set ω, the authority selects $x_1 \in_R \mathbb{Z}_p$ and sets $x_2 = x - x_1 \bmod p$. Furthermore, select a $(d-1)$-degree polynomial $q(\cdot)$ such that $q(0) = x_1$. Then, for each $i \in \omega$, choose $r_i \in_R \mathbb{Z}_p$, and compute $d_{i0} = g_2^{q(i)}(g_1h_i)^{r_i}$ and $d_{i1} = g^{r_i}$. For the default attribute θ, compute $d_{\theta 0} = g_2^{x_2}(g_1h)^{r_\theta}$ and $d_{\theta 1} = g^{r_\theta}$ by choosing $r_\theta \in_R \mathbb{Z}_p$. The private key is $SK = (SK_1, SK_2)$ where $SK_1 = \{d_{i0}, d_{i1}\}_{i\in\omega}$ and $SK_2 = \{d_{\theta 0}, d_{\theta 1}\}$.

Encryption Phase. Based on the logical split of user's attribute private key, the default attribute θ should be embeded in each ciphertext to make the decryption successful. The encryption algorithm works as follows.

– $\mathsf{Encrypt}(M, \omega')$: To encrypt a message M with respect to an attribute set ω', select $s \in_R \mathbb{Z}_p$ and compute $C_0 = Me(g_1, g_2)^s$, $C_1 = g^s$, $E_\theta = (g_1h)^s$ and $E_i = (g_1h_i)^s$ for each $i \in \omega'$. Finally, output the ciphertext $CT = (\omega' \cup \{\theta\}, C_0, C_1, \{E_i\}_{i\in\omega'\cup\{\theta\}})$.

Outsourced Decryption Phase. The outsourced decryption consists of the following two algorithms.

– $\mathsf{O\text{-}Decrypt\text{-}Outsource}(CT, SK_1)$: Suppose that a ciphertext CT is encrypted with an attribute set ω'. After receiving the private key component SK_1 for attribute set ω sent from a user, D-CSP continutes to compute the partially decrypted ciphertext CT': after selecting $S \subseteq \omega' \cap \omega$ with $|S| = d$, it computes $\frac{\prod_{i\in S} e(C_1, d_{i0})^{\Delta_{i,S}(0)}}{\prod_{i\in S} e(d_{i1}, E_i)^{\Delta_{i,S}(0)}} = e(g, g_2)^{sx_1}$. Fianlly, the partially decrypted ciphertext CT' is obtained as $CT' = (C_0, C_1, E_\theta, e(g, g_2)^{sx_1})$.
– $\mathsf{O\text{-}Decrypt\text{-}Dec}(CT', SK)$: Upon receiving CT' from D-CSP, the user completely decrypts the ciphertext and gets a message M as $M = \frac{e(d_{\theta 1}, E_\theta)C_0}{e(g, g_2)^{sx_1} e(C_1, d_{\theta 0})}$.

4.4 Security Analysis

The main challenge in our construction is to prevent attacks from the collusion between users and D-CSP. However, such collusion is resistant due to the random split on master key x for each user. More precisely, if two different users call for their private keys, AA will choose two random splits (x_1, x_2) and (x_1', x_2') such that $x_1 + x_2 = x \bmod p$ and $x_1' + x_2' = x \bmod p$. Note that x_1 and x_1' are used to generate the private key component SK_1 and SK_1' respectively, while SK_2 and SK_2' are separately generated from x_2 and x_2'. In this sense, the ciphertext can be correctly decrypted only when SK_1 matches SK_2. Therefore, even if a group of curious users collude with D-CSP to obtain all SK_1, they cannot forge a valid private key for themselves to perform decryption successfully out of their scopes.

Since basic outsoured decryption is supported, we only need to consider the security against type-I adversary. Then, we have the following security result.

Theorem 2. *The basic OABE scheme with outsourced decryption is secure against chosen-plaintext attack in selective model under DBDH assumption.*

Proof. Assume there exists an adversary $\mathcal{A}$ breaks the proposed scheme, we can build a simulator $\mathcal{S}$ that uses $\mathcal{A}$ as a sub-algorithm to solve the DBDH problem (X, Y, Z, T) as follows. The simulator $\mathcal{S}$ runs $\mathcal{A}$ and receives a challenge attribute set ω^* from $\mathcal{A}$. $\mathcal{S}$ sets $g_1 = X, g_2 = Y$ and $h = g_1^{-1}g^{-\alpha}$ where $\alpha \in_R \mathbb{Z}_p$. For $i \in \omega^*$, it selects $\alpha_i \in_R \mathbb{Z}_p$ and sets $h_i = g_1^{-1}g^{\alpha_i}$. For $i \notin \omega^*$, it selects $\alpha_i \in_R \mathbb{Z}_p$ and sets $h_i = g^{\alpha_i}$. Finally, $\mathcal{S}$ sends the public parameter $PK = (g, g_1, g_2, h, h_1, \ldots, h_n)$ to $\mathcal{A}$, where n is the number of attributes in universe. $\mathcal{A}$ is provided two types of oracles as follows:

i) Upon receiving the private key component SK_1 request on ω, $\mathcal{S}$ checks whether the entry $(\omega, \cdot, SK_1)$ exists in T. If so return SK_1; otherwise, if $|\omega \cap \omega^*| < d$, $\mathcal{S}$ picks $x_2 \in_R \mathbb{Z}_p$ and defines three sets Γ, Γ' and S, where $\Gamma = \omega \cap \omega^*, |\Gamma'| = d-1, \Gamma \subseteq \Gamma' \subseteq \omega$ and $S = \Gamma' \cup \{0\}$. Then, for each $i \in \Gamma'$, compute $d_{i0} = g_2^{\tau_i}(g_1h_i)^{r_i}$ and $d_{i1} = g^{r_i}$ where $\tau_i, r_i \in_R \mathbb{Z}_p$. For each $i \in \omega \backslash \Gamma'$, set $r_i = -y\Delta_{0,S}(i) + r_i'$ by choosing $r_i' \in_R \mathbb{Z}_p$. Finally, compute $d_{i0} = g_2^{\sum_{j\in\Gamma'}\Delta_{j,S}(i)\tau_j - (x_2+\alpha_i)\Delta_{0,S}(i)}(g_1h_i)^{r_i'}$ and $d_{i1} = g_2^{-\Delta_{0,S}(i)}g^{r_i'}$. Otherwise (i.e. $|\omega \cap \omega^*| \geq d$), $\mathcal{S}$ picks $x_1 \in_R \mathbb{Z}_p$ and randomly selects a $(d-1)$-degree polynomial $q(\cdot)$ with $q(0) = x_1$. Then, for each attribute $i \in \omega$, $d_{i0} = g_2^{q(i)}(g_1h_i)^{r_i}$ and $d_{i1} = g^{r_i}$ where $r_i \in_R \mathbb{Z}_p$.

ii) Upon receiving a private key request on ω with $|\omega \cap \omega^*| < d$, $\mathcal{S}$ checks whether the entry $(\omega, SK, \cdot)$ exists in T. If so return SK; otherwise if the value x_2 for such entry has not been selected, $\mathcal{S}$ picks $x_2 \in_R \mathbb{Z}_p$ and the remaining simulation is similar to the first case (i.e. $|\omega \cap \omega^*| < d$) to obtain SK_1, and compute $SK_2 = (d_{\theta 0} = g_2^{x_2}(g_1h)^{r_\theta},\ d_{\theta 1} = g^{r_\theta})$, where $r_\theta \in_R \mathbb{Z}_p$. Finally, after adding (ω, SK, SK_1) into T, $\mathcal{S}$ returns $SK = (SK_1, SK_2)$.

Two challenge messages M_0 and M_1 are chosen by $\mathcal{A}$. The simulator $\mathcal{S}$ flips a fair binary coin ν and generates the ciphertext of M_ν as $CT^* = (\omega^* \cup \{\theta\}, M_\nu T, g^z, g^{-z\alpha}, \{g^{z\alpha_i}\}_{i\in\omega^*})$. Note that: i) If $\mu = 0$, then $T = e(g,g)^{xyz}$. Let $s = z$ and we have $C_0 = M_\nu T = M_\nu e(g,g)^{xyz} = M_\nu e(g_1, g_2)^z, C_1 = g^z, E_\theta = g^{-z\alpha} = (g_1 g_1^{-1} g^{-\alpha})^z = (g_1 h)^z$ and $E_i = g^{z\alpha_i} = (g_1 g_1^{-1} g^{\alpha_i})^z = (g_1 h_i)^z$ for $i \in \omega^*$. Therefore, the ciphertext is a random encryption of the message M_ν under the attribute set ω^*.

The above querying phase is repeated with the restriction that $\mathcal{A}$ cannot issue a private key request on ω with $\gamma_d(\omega, \omega^*) = 1$.

$\mathcal{A}$ outputs a guess ν' of ν. If $\nu' = \nu$, $\mathcal{S}$ outputs $\mu' = 0$ to indicate that it was given a DBDH-tuple; otherwise, it outputs $\mu' = 1$ to indicate it was given a random 4-tuple.

5 Improved OABE Construction for Efficient Access Control System

In this section, based on the basic OABE, we further propose two OABE constructions supporting outsourced key-issuing and fine-grained access control.

5.1 OABE Construction with Outsourced Key-Issuing

Notice that in our basic construction, any adversary possessing either SK_1 or SK_2 cannot extract any useful information from the ciphertext. Thus, we are able to outsource the operation of generating SK_1 to KG-CSP but remain computing SK_2 at AA. Considering on this, we propose an OABE construction with outsourced key-issuing and decryption. Since the other phases are identical to our basic construction, we only provide the outsourced key-issuing protocol as follows.

- $\mathsf{O\text{-}KeyGen\text{-}PreProc}(\omega, MK)$: The preprocessing algorithm in outsourced key-issuing protocol is run by AA. It picks $x_1 \in_R \mathbb{Z}_q$ and sets $x_2 = x - x_1 \bmod q$. Finally, output (OK, AK) where $OK = x_1$ and $AK = x_2$.
- $\mathsf{O\text{-}KeyGen\text{-}Outsource}(\omega, OK)$: The outsourcing algorithm is run by KG-CSP. It randomly selects a $(d-1)$-degree polynomial $q(\cdot)$ with $q(0) = x_1$, and computes $SK_1 = (\{d_{i0}, d_{i1}\}_{i\in\omega})$ where $d_{i0} = g_2^{q(i)}(g_1 h_i)^{r_i}, d_{i1} = g^{r_i}$ and $r_i \in_R \mathbb{Z}_p$.
- $\mathsf{O\text{-}KeyGen\text{-}PostProc}(SK_1, AK)$: The postprocessing algorithm is run by AA. It computes $SK_2 = (d_{\theta 0}, d_{\theta 1})$ where $d_{\theta 0} = g_2^{x_2}(g_1 h)^{r_\theta}, d_{\theta 1} = g^{r_\theta}$ and $r_\theta \in_R \mathbb{Z}_p$. Finally output $SK = (SK_1, SK_2)$.

We have shown that the proposed construction is resistant to the type-I adversary in Section 4.4. Therefore, it is only needed to prove its security under the attack launched by the type-II adversary. Intuitively, in order to decrypt ciphertext, the adversary has to recover $e(g_1, g_2)^s$. The adversary could utilize Lagrange interpolation on SK_1 and $C_1 = g^s$ from ciphertext to recover the desired value. This will result in $e(g, g_2)^{sx_1}$ but blinded by $e(g, g_2)^{sx_2}$ which cannot be removed unless the other component of private key SK_2 is used.

Thus, we can get the following security result based on the analysis above.

Theorem 3. *The proposed OABE construction with outsourced key-issuing and decryption is secure against chosen-plaintext attack launched by type-II adversary.*

5.2 OABE Construction for Fine-Grained Access Control

Though we describe our outsourcing key-issuing technique in the threshold ABE, it can be easily extended to be applied to the access tree-based KP-ABE scheme [14] to enable fine-grained access control.

The idea behind this extension is to build a hybrid tree $\mathcal{T}$ as shown in Fig. 2, where $\wedge$ and $\vee$ denote AND and OR gates respectively, and A_i denotes the attribute.

To facilitate working with the access tree, we define a few notations and functions as follows.

- num_x is the number of children of an interior node x. Therefore, if assuming y is the child of node x, we could denote $\mathsf{index}(y)$ as such number associated with the node y.
- k_x is the threshold value of an interior node x, specifically, when $k_x = 1$, the threshold gate at x is OR gate and when $k_x = num_x$, that is an AND gate.
- The function $\mathsf{parent}(x)$ returns the parent of the node x in the tree. $\mathsf{attr}(x)$ returns the attribute associated with the leaf node x.

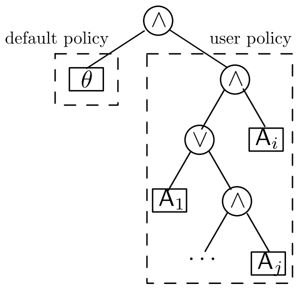


Fig. 2. Hybrid Tree Policy

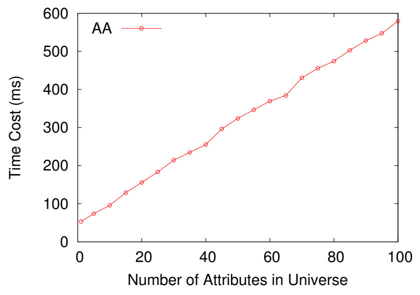
Suppose the access tree specified by user is denoted as $\mathcal{T}_{\mathrm{U}}$. Assuming the parameters have been assigned as the setup algorithm in Section 4.3, we provide the outsourced key-issuing protocol for access tree-based KP-ABE scheme as follows.

- $\mathsf{O\text{-}KeyGen\text{-}PreProc}(\mathcal{T}_{\mathrm{U}}, MK)$: Randomly pick a one-degree polynomial $q_R(\cdot)$ with $q_R(0) = x$. Set $x_1 = q_R(1)$ and $x_2 = q_R(2)$. Finally output $OK = x_1$ and $AK = x_2$.
- $\mathsf{O\text{-}KeyGen\text{-}Outsource}(\mathcal{T}_{\mathrm{U}}, OK)$: Firstly, choose a $(k_x - 1)$-degree polynomial $q_x(\cdot)$ for each node x (including leaves) in the tree $\mathcal{T}_{\mathrm{U}}$ in a top-down manner. We note that the polynomial $q_x(\cdot)$ is chosen with the restriction that $q_x(0) = x_1$ if x is the root node in $\mathcal{T}_{\mathrm{U}}$, otherwise $q_x(0) = q_{\mathsf{parent}(x)}(\mathsf{index}(x))$. Let Y_{U} be the set of leaf nodes in $\mathcal{T}_{\mathrm{U}}$, then the private key component SK_1 is set to be $(\{g_2^{q_y(0)}(g_1 h_{\mathsf{attr}(y)})^{r_y}, g^{r_y}\})$.
- $\mathsf{O\text{-}KeyGen\text{-}PostProc}(AK, SK_1)$: After generating the private key component $SK_2 = (\{g_2^{x_2}(g_1 h)^{r_\theta}, g^{r_\theta}\})$ where $r_\theta \in_R \mathbb{Z}_p$, AA outputs the private key $SK = (SK_1, SK_2)$.

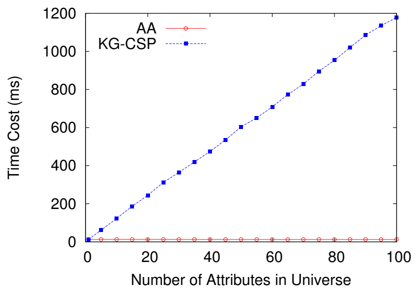
6 Performance Evaluation

As shown in Fig. 3, we provide a thorough experimental evaluation of the construction proposed in Section 5.1. Our experiment is simulated with the pairing-based cryptography (PBC) library [19] on a Linux machine with Intel Core 2 processor running at 2.40 GHz and 2G memory.

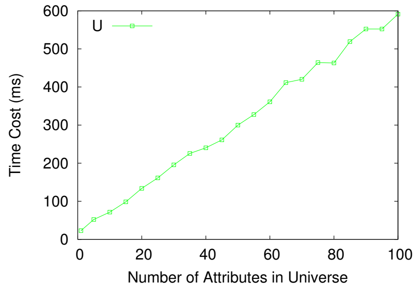
Our analysis is in terms of four phases in the construction. In Fig. 3(a) and Fig. 3(c), it is not surprising to see that as with existing ABE scheme [23], the computational cost in setup and encryption grows linearly with the number of attributes. In our outsourced construction, during the outsourced key-issuing phase the computation at AA just includes three exponentiations in $\mathbb{G}$, while that at U during the outsourced decryption phase just includes three bilinear pairings and one exponentiation in $\mathbb{G}_T$. The time cost for the both outsourced phases are reflected respectively in Fig. 3(b) and Fig. 3(d). Compared with the other OABE schemes such as [15,25], the computational cost at user side in the decryption algorithm is almost the same with ours. However, their work cannot support outsourced key-issuing because they used the blinding technique during user key-issuing.



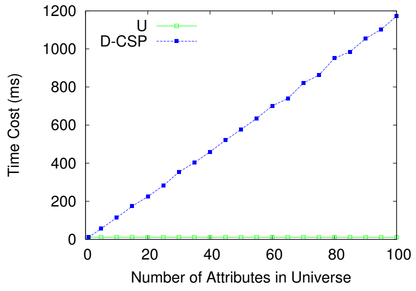
(a) Setup Phase



(b) Outsourced Key-Issuing Phase



(c) Encryption Phase



(d) Outsourced Decryption Phase

Fig. 3. Performance Evaluation

7 Related Work

Attribute-Based Encryption. The notion of ABE, which was introduced as fuzzy identity-based encryption in [23], was firstly dealt with by Goyal et al. [14]. Two different and complementary notions of ABE were defined as KP-ABE and CP-ABE. A construction of KP-ABE was provided in the same paper [14], while the first CP-APE construction supporting tree-based access structure in generic group model is presented by Bethencourt et al. [4].

Subsequently, a number of variants of ABE schemes have been proposed since its introduction. They range from extending its functionality to proposing schemes with stronger security proofs. Such as ABE schemes supporting for any kinds of access structures [21], ABE with multi-authorities [5], etc.

Recently, a novel paradigm for ABE was provided [15,25,18]. In [15], Green et al. considered to outsource the decryption of ABE to eliminate the overhead at user side, while an outsourced ABE with outsourced encryption and decryption was presented in [25][18]. We point out that the outsourcing decryption technique in [15,25] is to blind user's attribute private key by running a number of exponentiations. But such key blinded operation is eliminated in our construction in Section 4.3 through introducing a default attribute (actually, our technique provides a feasible way to realize the "piecewise key generation" property recently introduced in [22]). Moreover, it seems that all of the previous work lacks of the consideration on the reducing overhead computation at attribute authority. In another word, these work cannot support outsourced key-issuing due to the blinding technique used in the key generation algorithm.

Outsourcing Computation To reduce the load at local, it always desires to deliver expensive computational tasks outside. Actually, how to securely outsource different kinds of expensive computations has drawn much attention from theoretical computer science community [2,3,1,7]. But they are not suitable for reliving ABE computational overhead at user or authority side. To achieve this goal, the traditional approach is to utilize server-aided techniques [17,16,6]. However, previous work is oriented to accelerating the speed of exponentiation using untrusted servers. Directly utilizing these techniques in ABE will not work efficiently. Another approach might be to leverage recent general outsourcing technique or delegating computation [13,11,10,8,12] based on fully homomorphic encryption or interactive proof system. However, Gentry [12] has shown that even for weak security parameters on "bootstrapping" operation of the homomorphic encryption, it would take at least 30 seconds on a high performance machine. Therefore, even if the privacy of the input and output can be preserved by utilizing these general techniques, the computational overhead is still huge and impractical.

8 Conclusion

In this paper, we propose an efficient attribute-based access control system in cloud computing. In our system, two CSPs namely KG-CSP and D-CSP are introduced as employees to finish the outsourced heavy tasks for user management and file access respectively. The overhead at both users and attribute authority sides is thus being minimized. A challenging issue in the proposed system is how to outsource the computational task to CSPs without any private information leakage. To deal with this issue, we formulize an underlying primitive namely OABE and provide several OABE constructions with outsourced key-issuing and decryption. Finally, through extensive experiments, it demonstrates that our OABE construction achieves efficient key-issuing and decryption at AA and user sides respectively.

Acknowledgement. We are grateful to the anonymous referees for their invaluable suggestions. This work is supported by the National Natural Science Foundation of China (Nos. 61100224, 61272455 and 61272423), the National Key Basic Research Program of China (2013CB834204) and the Specialized Research Fund for the Doctoral Program of Higher Education of China (20100031110030). Besides, Lou's work was supported by US National Science Foundation under grant CNS-1155988.

References

1. Atallah, M.J., Frikken, K.B.: Securely outsourcing linear algebra computations. In: Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security, ASIACCS 2010, pp. 48–59. ACM, New York (2010)
2. Atallah, M.J., Pantazopoulos, K., Rice, J.R., Spafford, E.E.: Secure outsourcing of scientific computations. In: Zelkowitz, M.V. (ed.) Trends in Software Engineering, Advances in Computers, vol. 54, pp. 215–272. Elsevier (2002)
3. Benjamin, D., Atallah, M.J.: Private and cheating-free outsourcing of algebraic computations. In: Proceedings of the 2008 Sixth Annual Conference on Privacy, Security and Trust, PST 2008, pp. 240–245. IEEE Computer Society, Washington, DC (2008)
4. Bethencourt, J., Sahai, A., Waters, B.: Ciphertext-policy attribute-based encryption. In: IEEE Symposium on Security and Privacy 2007, pp. 321–334 (May 2007)
5. Chase, M., Chow, S.S.: Improving privacy and security in multi-authority attribute-based encryption. In: Proceedings of the 16th ACM Conference on Computer and Communications Security, pp. 121–130. ACM, New York (2009)
6. Chen, X., Li, J., Ma, J., Tang, Q., Lou, W.: New algorithms for secure outsourcing of modular exponentiations. In: Foresti, S., Yung, M., Martinelli, F. (eds.) ESORICS 2012. LNCS, vol. 7459, pp. 541–556. Springer, Heidelberg (2012)
7. Chen, X., Li, J., Susilo, W.: Efficient fair conditional payments for outsourcing computations. IEEE Transactions on Information Forensics and Security 7(6), 1687–1694 (2012)
8. Chung, K.-M., Kalai, Y.T., Liu, F.-H., Raz, R.: Memory delegation. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 151–168. Springer, Heidelberg (2011)
9. Fujisaki, E., Okamoto, T.: Secure integration of asymmetric and symmetric encryption schemes. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 537–554. Springer, Heidelberg (1999)
10. Gennaro, R., Gentry, C., Parno, B.: Non-interactive verifiable computing: Outsourcing computation to untrusted workers. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 465–482. Springer, Heidelberg (2010)

11. Gentry, C.: Fully homomorphic encryption using ideal lattices. In: Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, pp. 169–178. ACM, New York (2009)
12. Gentry, C., Halevi, S.: Implementing gentry's fully-homomorphic encryption scheme. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 129–148. Springer, Heidelberg (2011)
13. Goldwasser, S., Kalai, Y.T., Rothblum, G.N.: Delegating computation: interactive proofs for muggles. In: Proceedings of the 40th Annual ACM Symposium on Theory of Computing, STOC 2008, pp. 113–122. ACM, New York (2008)
14. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: Proceedings of the 13th ACM Conference on Computer and Communications Security, pp. 89–98 (2006)
15. Green, M., Hohenberger, S., Waters, B.: Outsourcing the decryption of abe ciphertexts. In: Proceedings of the 20th USENIX conference on Security, SEC 2011, p. 34. USENIX Association, Berkeley (2011)
16. Hohenberger, S., Lysyanskaya, A.: How to securely outsource cryptographic computations. In: Kilian, J. (ed.) TCC 2005. LNCS, vol. 3378, pp. 264–282. Springer, Heidelberg (2005)
17. Jakobsson, M., Wetzel, S.: Secure server-aided signature generation. In: Kim, K. (ed.) PKC 2001. LNCS, vol. 1992, pp. 383–401. Springer, Heidelberg (2001)
18. Li, J., Jia, C., Li, J., Chen, X.: Outsourcing encryption of attribute-based encryption with mapreduce. In: Chim, T.W., Yuen, T.H. (eds.) ICICS 2012. LNCS, vol. 7618, pp. 191–201. Springer, Heidelberg (2012)
19. Lynn, B.: The pairing-based cryptography library, http://crypto.stanford.edu/pbc/
20. McDaniel, P., Prakash, A.: Methods and limitations of security policy reconciliation. ACM Trans. Inf. Syst. Secur. 9(3), 259–291 (2006)
21. Ostrovsky, R., Sahai, A., Waters, B.: Attribute-based encryption with non-monotonic access structures. In: Proceedings of the 14th ACM Conference on Computer and Communications Security, CCS 2007, pp. 195–203. ACM, New York (2007)
22. Sahai, A., Seyalioglu, H., Waters, B.: Dynamic credentials and ciphertext delegation for attribute-based encryption. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 199–217. Springer, Heidelberg (2012)
23. Sahai, A., Waters, B.: Fuzzy identity-based encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (2005)
24. Yu, S., Wang, C., Ren, K., Lou, W.: Achieving secure, scalable, and fine-grained data access control in cloud computing. In: Proceedings of the 29th Conference on Information Communications, INFOCOM 2010, pp. 534–542. IEEE Press, Piscataway (2010)
25. Zhou, Z., Huang, D.: Efficient and secure data storage operations for mobile cloud computing. Cryptology ePrint Archive, Report 2011/185 (2011)