Traitor Tracing Based on Partially-Ordered Hierarchical Encryption

Yan Zhu(✉), Dandan Li, and Liguang Yang

School of Computer and Communication Engineering,
University of Science and Technology, Beijing 100083, China
zhuyan@ustb.edu.cn

Abstract. Recently, more and more enterprises and individuals have moved their data into the cloud. To meet this practical requirement, this paper addresses how to establishes a bridge between role-based access control (RBAC) and cloud storage in order to fully preserve investment in existing RBAC systems. We present a new scheme for secure migrating the resources from RBAC systems to cloud storage. This scheme takes full advantage of RBAC, which provides a well-designed and easy-to-manage approach for accessing cloud resources without user intervention. This scheme, called Partially-ordered Hierarchical Encryption (PHE), which implements the partial-order key hierarchy, similar to role hierarchy in RBAC, in public-key infrastructure. In addition, this construction provides traitor tracing to support efficient digital forensics. The performance analysis shows that our construction has following features: dynamic joining and revoking users, constant-size ciphertexts and decryption keys, and lower overloads for large-scale systems.

Keywords: Security · Encryption · Cloud storage · Partial order · Key hierarchy · Traitor tracing

1 Introduction

In recent years, more and more enterprises and individuals have moved their data, such as personal data and large archive system, into the cloud. Cloud-based storage could be particularly attractive for consumers by providing on demand capacity, low-cost service, and long-term archive. Furthermore, cloud services have brought great convenience to people's lives because consumers can access applications and data from the cloud anywhere in the world on demand.

However, there exist some obstacles for migrating the resources in information systems, especially for an amount of existing RBAC systems, into the public cloud. One of these obstacles is the security of migrated resources. Several recent surveys [1] show that 88% potential cloud consumers worry about the privacy of their data, and security is often cited as the top obstacle for cloud adoption. Unfortunately, traditional security mechanisms, such as access control technology, are not suitable for the cloud environment due to the outsourcing-service

M. Yung et al. (Eds.): INTRUST 2014, LNCS 9473, pp. 278–293, 2016.
DOI: 10.1007/978-3-319-27998-5_18

characteristics of cloud storage and the untrusted or honest-but-curious assumption of cloud service providers. On the other hand, the protection of the outsourced data against illegal redistribution via traitor's illegal decoders (or illegal decryption softwares) has become increasingly important due to huge potential commercial value of data stored in cloud.

In order to solve this issue, attribute-based encryption (ABE) [2–6] has been proposed in the recent years. Although ABE is a powerful tool which meets a variety of application requirements, the current ABE schemes cannot fulfill the requirements for the existing RBAC systems owing to lack of support for partial ordering relations. It is well-known that RBAC is an industry recognized and widely adopted access control model. In this model, role hierarchy (RH) is an important notion, which reflects organization's lines of authority and responsibility. Mathematically, role hierarchies are partial orders. Unfortunately, this kind of partial ordering relation still cannot be implemented in the existing ABEs. Therefore, it is necessary to develop a new RBAC-compatible encryption scheme to support the secure migration from RBAC systems into the cloud.

To construct a cryptosystem compatible with RBAC model [7], several schemes for hierarchical key management (HKM) have been designed [8,9]. These existing schemes have following common features: 1) each user has a secret-key sk_i corresponding to a role c_i in RH; 2) there exists an efficient way to derive a descendant's key sk_j from the own key sk_i in accordance with the partial order relation $c_j \preceq c_i$ in RH; and 3) key derivation can be implemented under the precondition of the existence of an one-way function.

Existing schemes can effectively derive the keys with the help of partial order structure. However, such kind of derivation process has following problems:

- A role may be assigned to multiple users who share the same secret-key. That means there is no way to distinguish those assigned users; and
- The secret-key derivation is not able to support additional function, such as the traitor tracing, in terms of digital forensics for group-oriented cryptosystem.

To address these problems, it is necessary to design a construction for hierarchical cryptosystems, considering the new features provided by some recently proposed cryptography technologies, such as IBE [2], HIBE [10] and ABE [11]. In such a construction, a user secret-key must be unique and is accompanied by the user identity. In addition, the derivation of secret-key in such a construction should be avoided. To this end, we introduce a new hierarchical key structure using the public-key settings. Our construction can achieve following functions:

- Each role is assigned with a public-key (called role-key) in RBAC, and there exists a derivation function on these public-keys in accordance with RH;
- Each user has a unique identity and private key, which retain his/her role information, but the derivation of secret-key is prohibited; and
- Such a key structure can be used to establish some important security mechanisms, such as encryption, signature, revocation, and traitor tracking.

One compelling advantage of our key structure is that it can be seamlessly integrated into the existing RBAC systems. Consequently, an RBAC system can directly use the public role-key to encrypt resources in terms of users' assigned roles, and then the users owned the senior roles can use their privacy-keys to decrypt the encrypted resources. This kind of cryptosystem can be used for secure migrating the resources from existing RBAC systems to cloud. Other potential applications of our solution include email encryption system (EES), privacy preservation for peer-to-peer (P2P) data sharing, and encrypted file system (EFS).

Table 1. Comparison of several key management methods with user management [a]

	Stateful schemes	Stateless schemes		
	LKH [12]	CS [13]	LSD [14]	Our Works
Cryptography settings	symmetric-key	symmetric-key	symmetric-key	pubic-key
User key storage	$O(\log n)$	$O(\log n)$	$O(\log^{1+\epsilon} n)^b$	$O(1)$
Encryption cost	$O(n^{1/k})^c$	$O(\log \log n)$	$O(\log n)$	$O(t + \frac{n}{m})$
Average bandwidth	$O(t \log(n/t))$	$O(t \log(n/t))$	$O(t)$	$O(t + \frac{n}{m})$ fixed
Worst case bandwidth	$\min(t \log \frac{n}{t} + t, n - t)$	$\min(t \log \frac{n}{t}, n - t)$	$\min(4t - 2, n - t)$	$t + \frac{n}{m}$ fixed
Traitor tracing	$O(\log n)$	$O(\log n)$	$O(t \log(n/t))$	$O(\log(\frac{n}{m}) + \frac{m}{t})^d$
Key-updating complexity	high	moderate	low	not modify

where, [a] n is the total number of users, t is the number of revoked devices, and m is the average number of users in a subset. [b] ϵ is any number > 0. [c] k is a parameter which mean the number of stratified subsets to obtain a reasonable computation cost, i.e., when n is less than one trillion, $n^{1/8} < \log n$. [d] references the preference evaluation.

Our Contributions. In this paper, our objective is to establishes a bridge between RBAC and secure cloud storage in order to fully preserve investment in existing RBAC systems. To meet this goal, our core task is construct an effective RBAC-compatible cryptosystem for cloud data encryption. This kind of cryptosystem takes full advantage of RBAC, which provides a well-designed and easy-to-manage approach for accessing cloud resources without user intervention. To achieve our task, we present a new cryptosystem, called as Partially-ordered Hierarchical Encryption (PHE) with traitor tracing. The major contributions of this work are summarized as follows:

– We provided a practical Partially-ordered Hierarchical Encryption (PHE) construction, which not only has semantic security and secure key hierarchy, but also supports following features: stateless receivers, dynamic granting, tight security, and a large number of users;

– We given a full security analysis of our cryptosystem, including semantical security under chosen plaintext attacks. More important, our scheme satisfied a new security definition of key management, called secure key hierarchy, against privilege attack and access attack; and
– We provided traitor tracing mechanism based on key hierarchy, which has great practical significance to preserve the integrity and validity of long-term cryptosystems and to prevent the leakage of cloud outsourced data via illegal decoders (or illegal decryption softwares).

In addition, our PHE scheme provides several new secure features, such as public user label, constant-size user key storage, $O(\log(n))$ tracing, lower computational costs and communication bandwidths.

Table 1 shows a comparison of our scheme and some broadcast encryption schemes including Logical Key Hierarchy(LKH) [12], Complete Subtree(CS) [13], Subset Difference(SD) [15], and Layered Subset Difference(LSD) [14]. Although some existing public-key schemes have adopted the hierarchical structure, this comparison does not consider them due to the reason that they do not have a unique key assigned to eash user, and therefore cannot achieve the features of traitor-tracking. From Table 1, it is obvious that the performance of our scheme is substantially better than existing methods with respect to transmission, storage, computation, and traitor tracing costs.

Organization. The rest of the paper is organized as follows. Section 2 describes the research background and the definition of key structure. In Sect. 3, we address our PHE scheme for cryptographic access control on RBAC. Section 4 describes the traitor tracing mechanism, for digital forensics. The results of security analysis is showed in Sect. 5, respectively. We summary the related workin Sect. 6. We conclude and discuss the future work in Sect. 7.

2 Background and Definition

Given a secure key hierarchy $\Psi = \langle C, E, K \rangle$ and the total number n of classes, we can define a (t, n)-Partially-ordered Hierarchical Encryption (PHE), which ensures a content provider to securely transmit a message to a subset of authorized users under the assumption of at most t collusion. More formally, a (t, n)-PHE scheme with a security parameter s is a 6-tuple of probabilistic algorithms $(Setup, Join, Encrypt, Decrypt, Trace)$ described as follows:

1. *Setup*(Ω, s, t)*:* Takes as input a partial-order hierarchy Ω, a security parameter s and a maximal collusion number t. It outputs a main encryption key pk_0 as the starting point of cryptosystem, a set of public parameters P[1], and a master key mk as the manager secret.
2. *Join*$(P, mk, c_i \text{ or } u_{i,j})$*:* Includes two sub-algorithms:

[1] The signature of P can be generated avoid tampering.

  - *Join*(P, mk, c_i)*:* Takes as input the manager secret mk and a group identifier c_i. It generates an encryption key pk_i and some public parameters pp_i as the description of this class. $P = P \cup \{pp_i\}$ is made public.
  - *Join*$(P, mk, u_{i,j})$*:* Takes as input the manager secret mk and a user identifier $u_{i,j}$. It outputs a user key $sk_{i,j} = (lab_{i,j}, dk_{i,j})$. $P = P \cup \{lab_{i,j}\}$ and $sk_{i,j}$ is sent to $u_{i,j}$ securely.
3. *Encrypt*(P, pk_i, M)*:* Encrypts a message M using the public key pk_i and outputs a ciphertext $\mathcal{C}_i$.
4. *Decrypt*$(P, sk_{i,j}, \mathcal{C}_k)$*:* Decrypts a ciphertext $\mathcal{C}_k$ using a decryption keys $dk_{i,j}$ and outputs the message M, if $u_{i,j} \in c_i$ and $c_i \preceq c_k$.
5. *Trace*$^{\mathcal{D}}(P, pk, mk)$*:* Suppose an adversary uses k user keys $R = \{sk_{i_1,j_1}, \cdots, sk_{i_k,j_k}\}$ to create a decryption box $\mathcal{D}$. As an oracle algorithm on $\mathcal{D}$, it takes as input pk, mk, and can determine at least one key in the collusion R.

A tracing algorithm is said to be 'Black Box' if the decoder $\mathcal{D}$ can only be queried as an Oracle but not opened to reveal its internal keys. The scheme is said to be 't-resilient' if there is an effective cryptosystem with the collusion of at most t keys. Note that, the four algorithms $(Setup, Join, Encrypt, Decrypt)$ are used to realize basic cryptographic access control under RBAC model, and the algorithms $Trace$ provide traitor tracing for digital forensics.

3 PHE Scheme for Access Control

3.1 Proposed PHE Scheme

Given a secure key hierarchy $\Omega = \langle C, E \rangle$, a security parameter s, and the maximal coalition size t. Let G_q be a group of prime order q and $\log_2 q > s$. One can take as G_q the subgroup of $\mathbb{Z}_p^*$ of order q, where p is a large prime with $q|p-1$. Let $g \in_R \mathbb{Z}_p^*$ be a generator of G_q.

1. *Setup*(Ω, s, t)*:* The manager chooses t random integers $a_1, \cdots, a_t \in \mathbb{Z}_q^*$ to construct a random polynomial $f(x) = \sum_{i=1}^{t} a_i x^i \pmod q$ with degree t. It therefore randomly chooses t integers $x_1, \cdots, x_t$ to generate $(x_1, f(x_1)), \cdots, (x_t, f(x_t))$. It makes the parameters $P = \{p, q, (x_1, g^{f(x_1)}), \cdots, (x_t, g^{f(x_t)})\}$ public. Without loss of generality, we assume that c_0 be only the senior-most class in Ω. It chooses a random integer $s_0 \in_R \mathbb{Z}_q$ for c_0 as its secret, so that the random polynomial $f(x)$ is replaced by $p_0(x) = s_0 + \sum_{i=1}^{t} a_i x^i \pmod q$. It then uses $(x_1, p_0(x_1)), \cdots, (x_t, p_0(x_t))$ to generate an initial encryption key:

$$\begin{aligned} pk_0 &= \langle g, z_{0,0}, (x_1, z_{0,1}), \cdots, (x_t, z_{0,t}), T_0 \rangle \\ &= \langle g, g^{p_0(0)}, (x_1, g^{p_0(x_1)}), \cdots, (x_t, g^{p_0(x_t)}), \emptyset \rangle \end{aligned} \tag{1}$$

   where, $z_{0,i} = g^{p_0(x_i)} = g^{s_0} \cdot g^{f(x_i)} \pmod p$ is computed from P and $T_0 = \emptyset$ denotes a null initial control domain. The system manager keeps $mk = \{s_0, a_1, \cdots, a_t\}$ secret.

2. *Join(P, mk, c_i or $u_{i,j}$):* which includes two forms:
   (a) *Join(P, mk, c_i):* To generate pk_i and pp_i of $c_i \in C$, the manager assigns the random $s_i \in \mathbb{Z}_q$ for c_i as its secret. For $\forall c_l \in C$ and $c_i \prec_d c_l$, it computes $t_{i,l} = g^{(s_l - s_i)} \pmod p$ as the public parameter of this relation, and then defines $pp_i = \{t_{i,l}\}_{c_i \prec_d c_l}$ as the set of all relations which directly dominate c_i. Finally, it appends s_i and pp_i into mk and P respectively, i.e., $mk = mk \cup \{s_i\}$ and $P = P \cup pp_i$.
   The encryption key pk_i in c_i can be computed from the polynomial $p_i(x) = s_i + f(x)$. In terms of mk and P, the manager has

$$\begin{aligned} pk_i &= \langle g, z_{i,0}, (x_k, z_{i,k})_{k=1}^t, T_i \rangle \\ &= \langle g, g^{p_i(0)}, (x_k, g^{p_i(x_k)})_{k=1}^t, T_i \rangle, \end{aligned} \tag{2}$$

   where, T_i is a set of all relations in $\uparrow c_i$, i.e., $T_i = \{t_{j,l}\}_{c_j, c_l \in \uparrow c_i, c_j \prec_d c_l}$.
   (b) *Join($P, mk, u_{i,j}$):* To generate $sk_{i,j}$ of $u_{i,j}$, the manager computes the random polynomial $p_i(x) = s_i + f(x) \pmod q$ by using the secret in mk. It generates a new random integer $x_{i,j} \in_R \mathbb{Z}_q$ and sends $sk_{i,j} = (x_{i,j}, p_i(x_{i,j}))$ to the user via a secret channel, where $lab_{i,j} = x_{i,j}$, $dk_{i,j} = p_i(x_{i,j})$, and $P = P \cup \{lab_{i,j}\}$.
3. *Encrypt(P, pk_i, M):* For a session key $ek \in G_q$[2], the user randomly chooses a random number $r \in_R \mathbb{Z}_q$, and then computes the ciphertext by pk_i as follows:

$$\begin{aligned} \mathcal{C}_i &= \langle h, S_i, (x_k, h_{i,k})_{k=1}^t, T_i' \rangle \\ &= \langle g^r, ek \cdot z_{i,0}^r, (x_k, z_{i,k}^r)_{k=1}^t, \{t'_{k_1,k_2}\}_{t_{k_1,k_2} \in T_i} \rangle. \end{aligned} \tag{3}$$

   where, $h_{i,k} = z_{i,k}^r \pmod p$, $t'_{k_1,k_2} = t_{k_1,k_2}^r$, and $T_i' = \{t_{k_1,k_2}^r\}_{t_{k_1,k_2} \in T_i}$ denotes a control domain which includes all relations in $\uparrow c_i$.
4. *Decrypt($P, sk_{i,j}, \mathcal{C}_l$):*
   After receiving a cipher-text $\mathcal{C}_l = \langle h, S_l, (x_k, h_{l,k})_{k=1}^t, \{t'_{k_1,k_2}\}_{t_{k_1,k_2} \in T_l} \rangle$, the user computes the following equation by the private key $sk_{i,j} = \langle x_{i,j}, y_{i,j} \rangle$ if we hold $u_{i,j} \in c_i$, $c_i \preceq c_l$, and

$$U_{\mathcal{C}_l}(sk_{i,j}) = \frac{h^{y_{i,j} \cdot \lambda_0(x_{i,j})} \prod_{k=1}^t h_{l,k}^{\lambda_k(x_{i,j})}}{(\prod_{c_{k_1} \prec_d c_{k_2} \in \Delta(l,i)} t'_{k_1,k_2})^{\lambda_0(x_{i,j})}}, \tag{4}$$

   where, $\lambda_k(x_{i,j}) = \prod_{l=0, l \neq k}^t \frac{x_l}{x_l - x_k} \pmod q$ is the coefficient of Lagrange interpolation polynomial[3] for $\{x_0 = x_{i,j}, x_1, \cdots, x_t\}$, and $\Delta(l,i) = \{c_{k_1} \prec_d c_{k_2} : c_{k_1}, c_{k_2} \in \Gamma(l,i)\}$ denotes the set of direct dominations on an arbitrary path between c_i and c_l. It therefore can obtain the plaintext $ek = S_i / U_{\mathcal{C}_l}(sk_{i,j})$.

[2] The plaintext (ek or M) must be converted into an element of G_q, see ElGamal encryption system.

[3] Given a set of $t+1$ different data points $(x_0, y_0), \cdots, (x_t, y_t)$, the language interpolation polynomial is a linear combination $L(x) = \sum_{j=0}^t y_j \lambda_j(x)$ where the coefficient $\lambda_j(x) = \prod_{i=0, i=j}^t \frac{x - x_i}{x_j - x_i}$. Here, we set $x = 0$ to compute $L(0)$.

Before going further, we briefly show that the encryption scheme is valid by

$$\begin{aligned} U_{\mathcal{C}_l}(sk_{i,j}) &= \frac{g^{p_i(x_{i,j})\cdot\lambda_0(x_{i,j})\cdot r}\prod_{k=1}^{t} g^{p_l(x_k)\cdot\lambda_k(x_{i,j})\cdot r}}{(\prod_{c_{k_1}\prec_d c_{k_2}\in\Delta(l,i)} t_{k_1,k_2}^{r})^{\lambda_0(x_j)}} \\ &= \frac{g^{p_i(x_{i,j})\cdot\lambda_0(x_{i,j})\cdot r}\prod_{k=1}^{t} g^{p_l(x_k)\cdot\lambda_k(x_{i,j})\cdot r}}{g^{\sum_{c_{k_1}\prec_d c_{k_2}\in\Delta(l,i)}(s_{k_2}-s_{k_1})\cdot\lambda_0(x_{i,j})\cdot r}} \\ &= \frac{g^{p_i(x_{i,j})\cdot\lambda_0(x_{i,j})\cdot r}\prod_{k=1}^{t} g^{p_l(x_k)\cdot\lambda_k(x_{i,j})\cdot r}}{g^{(s_i-s_l)\cdot\lambda_0(x_{i,j})\cdot r}} \\ &= g^{p_l(x_{i,j})\cdot\lambda_0(x_{i,j})\cdot r}\prod_{k=1}^{t} g^{p_l(x_k)\cdot\lambda_k(x_{i,j})\cdot r} \\ &\overset{x_0=x_{i,j}}{=} g^{\sum_{k=0}^{t} p_l(x_k)\cdot\lambda_k(x_0)}\cdot r = g^{p_l(0)\cdot r} = z_{l,0}^{r}. \end{aligned} \tag{5}$$

where $s_i - s_l = \sum_{c_{k_1}\prec_d c_{k_2}\in\Delta(l,i)}(s_{k_2}-s_{k_1}) \pmod q$ for an arbitrary path $\Gamma(l,i)$ between c_i and c_l[4], and $p_l(x_{i,j}) = s_l + f(x_{i,j}) = p_i(x_{i,j}) - (s_i - s_l) \pmod q$.

3.2 Further Discussion

In fact, the above process is also constructed from bottom (junior-class) to top (senior-class). In the case of many senior-most classes, the *Setup* algorithm is still available. Without loss of generality, we assume that $c_0^{(1)}, c_0^{(2)}, \cdots, c_0^{(l)}$ are l senior-most classes in Ω. Then, it chooses a random integer $s_0^{(i)} \in_R \mathbb{Z}_q$ for $c_0^{(i)}$ as the secret of this class, such that it constructs l random polynomials, $p_0^{(i)}(x) = s_0^{(i)} + \sum_{k=1}^{t} a_k x^k$, where $i \in [1,l]$. Finally, the encryption key is generated:

$$\begin{aligned} pk_0^{(i)} &= \langle g, z_{0,0}^{(i)}, (x_1, z_{0,1}^{(i)}), \cdots, (x_t, z_{0,t}^{(i)}), T_0\rangle \\ &= \langle g, g^{p_0^{(i)}(0)}, (x_1, g^{p_0^{(i)}(x_1)}), \cdots, (x_t, g^{p_0^{(i)}(x_t)}), \emptyset\rangle, \end{aligned}$$

where, $g^{p_0^{(i)}(x_k)} = g^{s_0^{(i)}} g^{f(x_k)} \pmod p$.

In order to share information, the encryption keys pk_n of junior-most classes are usually made public, which is called the main encryption key, e.g., for a enterprise management system, if the encryption key of "Engineering Dept" class is used to send the message, all employees are able to decrypt it by their own private keys. Moreover, the storage ratio of encryption keys is also an important feature considering a number of classes in the large-scale organizations. We, of course, expect that it is as low as possible. Since $p_i(x) = (s_i - s_l) + p_l(x)$, the user can generate pk_i by using a known pk_j and public parameters P for $i \neq j$. For example, the user can compute her/his own encryption key pk_i from a junior-most encryption key pk_n by $\dot{T}_i = \prod_{c_j\prec_d c_l\in\Delta(n,i)} t_{j,l} = g^{\sum_{c_j\prec_d c_l\in\Delta(n,i)}(s_l - s_j)} = g^{s_i - s_n} \pmod p$ and

[4] For the different pathes, we have the same polynomial $p_i(x) = s_i + \sum_{i=1}^{t} a_i x^i$, because $p_i(x) = (s_i - s_{i-1}) + (s_{i-1} - s_{i-2}) + \cdots + (s_1 - s_l) + p_l(x)$ for any path $s_i, s_{i-1}, \cdots, s_1, s_l$.

$$\begin{aligned} pk_i &= \langle g, z_{i,0}, (x_k, z_{i,k})_{k=1}^{t}, T_i \rangle \\ &= \langle g, z_{n,0} \cdot \dot{T}_i, (x_k, z_{n,k} \cdot \dot{T}_i)_{k=1}^{t}, T_i \rangle, \end{aligned} \tag{6}$$

where, $z_{n,k} \cdot \dot{T}_i = g^{p_n(x_k)} \cdot g^{s_i - s_n} = g^{p_i(x_k)} \pmod{p}$, and T_i is found from P in terms of Eq. (2). Therefore, the user only needs to store an encryption key pk_i and a private key $sk_{i,j} = (label_{i,j}, dk_{i,j})$.

The key hierarchy is saved in public parameters P, irrespective of the user private keys, so that the public parameters can be merely modified dynamically to support the change of the key hierarchy.

4 PHE Scheme for Traitor Tracing

It is very hard for the adversary to directly break a cryptosystem with provable security, but the adversary could make other means to break it. It is well-known that "the easiest way to capture a fortress is from within". Based on the same idea, the collusion attack between the adversary and some corrupted users (called traitors) is such an internal attack for group-oriented cryptosystem. In this attack, the adversary may have access to a set of legitimate user's secret keys to decrypt the ciphertext. In order to withstand such attacks, traitor tracing is introduced in the recent years. Usually, the traitor tracing algorithm is an effective detection approach to find out the corrupted users from a group of authorized users based on a found pirate decoder. We prefer that the tracing algorithm is only able to access any pirate decoder as a black box and perform the tracing based on the decoder's response on different input ciphertexts.

The traitor tracing is an efficient mechanism to support digital forensics in the existing group-oriented cryptosystems. Some tracing schemes have also been proposed via the polynomial interpolation method in the recent years. We here propose a new traitor tracing scheme for our partial-order key hierarchy on the basis of these existing schemes. This algorithm only needs to know the public label $label_{i,j}$ of users rather than their private keys. Note that, traitor tracing, as a way of digital forensics, has a precondition where the adversary cannot forge an 'unused' key to avoid tracing. We will prove that this attack is infeasible for our scheme. We now turn our attention to the tracing algorithm from the following two aspects:

4.1 Single-Key Tracing

The single-key tracing algorithm focuses on finding the traitors of collusion one by one. It is easy to find that at most t users cannot forge a new unused key in the corrupted class, such that we can find all traitors only if we search all used keys in this class. For such a collusion attack, we can use the revocation-based algorithm to construct a ciphertext, revoked by the suspicious key, into the illegal decoder. If the decoder does not work, this revocation-based key includes at least

a traitor. Otherwise, we search other users in this subset. Finally, we can find all traitors. To improve the performance, we can check out t suspicious keys at the same time. Hence, the searching complexity is $O(m/t)$, where m is the total number of users in a security class or a group of users.

Many tracing algorithms [16] have noticed that a certain linear combination of $sk_1, \cdots, sk_m$ is also a 'new' private key, but in this case the adversary is not confined to the original decryption algorithm to build a decryption box. In such a case, this 'single-key' is not a new key but a linear combination of some keys. For such a decoder, we can construct an encryption key, which includes t user keys, and search all combinations among the keys in this subset. Hence, the searching complexity is $O(\binom{t}{m})$.

4.2 Hierarchical Tracing

The hierarchical tracing algorithm is a more efficient method to find the traitors in terms of partial-order key hierarchy. According to the property of threshold cryptosystem, our proposed scheme is a t-resilient encryption based on CDH assumption in the honest classes, showing that the traitors cannot collude to forge a new key outside the corrupted classes. This property gives us an advantage for constructing the tracing algorithm.

In contrast to single-key tracing, we can first go through each class c_i in a key hierarchy Ψ to locate the suspicious classes of the traitors, and then use single-key tracing algorithm to find the actual traitors in every class. In terms of this idea, given an illegal decoder, we present a black-box traitor tracing algorithm based on the key hierarchy, which involves two steps: *subtree searching* and *subset traversing*, as follows:

V1. *Subtree searching*: Given a key hierarchy Ψ, we start from $c_i \leftarrow c_n$ (the junior-most class) in C and run the following processes from bottom to top:
  S1. Randomly selects t unused shares $\langle x_1, x_2, \cdots, x_t \rangle$ and constructs an enabling block:

$$\mathcal{C}_i = \langle g^r, ek \cdot g^{rp_i(0)}, (x_k, g^{rp_i(x_k)})_{k=1}^{t}, \{t_{j,l}^r\}_{t_{j,l} \in T_i} \rangle.$$

  S2. Sends $\langle \mathcal{C}_i, E(ek, M) \rangle$ to the decoder.
  S3. If the decoder can return correctly the message M, we consider c_i as a suspicious class and run V1 by $c_i \leftarrow c_j$ for $\forall c_j \prec_d c_i$, otherwise, repeat V1 by a sibling node of c_i.

V2. *Subset traversing*: Let $\langle c_1', c_2', \cdots, c_k' \rangle$ be the set of suspicious subset by $V1$, for each c_i' in this set, we run the following processes:
  T1. Chooses any m user's labels in c_i' at random, $\{x_{i,1}, \cdots, x_{i,m}\}$, $m \leq t$, and then randomly selects $t - m$ unused shares, $\langle v_1, v_2, \cdots, v_{t-m} \rangle$, and constructs an enabling block:

$$\mathcal{C}_i' = \left\langle \begin{matrix} g^r, ek \cdot g^{rp_i(0)}, (x_{i,j}, g^{rp_i(x_{i,j})})_{j=1}^{m}, \\ (v_k, g^{rp_i(v_k)})_{k=1}^{t-m}, \emptyset \end{matrix} \right\rangle.$$

  T2. Sends $\langle \mathcal{C}_i', E(ek, M) \rangle$ to the pirate decoder.

T3. If the decoder does not output correctly M, we consider the set of label, $\{x_{i,1}, \cdots, x_{i,m}\}$, as a set of traitors and decrease the number of key of this set to run T1. Otherwise, repeats T1 until no more users.

Therefore, our tracing algorithm improves computation complexities and searching times as a result that key hierarchy divides the users into a large number of classes in the key hierarchy. Especially, in the worst case, the complexity of subtree searching is $O(\log n)$ time queries, where n is the number of classes.

5 Security Analysis

We define the security of PHE scheme in terms of a family of security games between a challenger and an adversary. The partial-order hierarchy Ω and system parameters P are fixed, and the adversary is allowed to depend on them. The users can be divided into two categories: the honest users and the corrupted users, so that a set of corrupted users $\mathcal{R}$ is built. The responsive classes is called as honest classes C_1 or corrupted classes C_2, in which the corrupted users can access all encrypted messages. Sometimes, there exist many honest and corrupted users in the same class. We first define a general model against collusion attacks:

1. Initial: The challenger $\mathcal{B}$ constructs an arbitrary partial-order hierarchy Ω, and then runs $Setup(\Omega, s, t)$ to generate the partial-order key hierarchy Ψ and initial public parameters P, and sends them to the adversary $\mathcal{A}$.
2. Learning: $\mathcal{A}$ adaptively issues n times queries $q_1, \cdots, q_n$ to learn the information of Ψ, where q_i is one of the following:
   - Honest class/user query ($u_{i,j} \notin \mathcal{R}$): using $Join(P, mk, c_i$ or $u_{i,j})$, $\mathcal{B}$ generates a class/user label (pp_i, pk_i, $lab_{i,j}$) and sends $lab_{i,j}$ to $\mathcal{A}$.
   - Corrupted class/user query ($u_{i,j} \in \mathcal{R}$): $\mathcal{B}$ generates a class (pp_i, pk_i) with the corrupted users, or a user label $lab_{i,j}$ and a decryption key $dk_{i,j}$, and returns $(lab_{i,j}, dk_{i,j})$ to $\mathcal{A}$.

   $\mathcal{A}$ ends up with a key hierarchy Ψ (include P, pk_i) and a collusion set $\{sk_{i,j}\}_{u_{i,j} \in \mathcal{R}}$. Note that the decryption query is unnecessary because $\mathcal{A}$ can use the corrupted key to generate it.
3. Challenge: $\mathcal{A}$ chooses two equal length plaintexts M_0, $M_1 \in \mathbb{M}$ and appoints a classes c_i on which it wishes to be challenged. $\mathcal{B}$ picks a random bit $b \in \{0, 1\}$ and sends the challenge ciphertext $\mathcal{C}_i = Encrypt(P, pk_i, M_b)$ or $Revoke(P, pk_i, M_b, R_i)$ to $\mathcal{A}$. where, $\mathcal{R}_i$ denotes all corrupted users in $\uparrow r_i$.
4. Guess: $\mathcal{A}$ outputs a guess $b' \in \{0, 1\}$. $\mathcal{A}$ wins if $b = b'$, and otherwise it loses.

There are several important variants for this game:

- In a game for chosen plaintext attack (CPA), the adversary $\mathcal{A}$ may not issue the corrupted user queries and decryption queries during the learning phase.

– In a game for user's private key attack, the challenger $\mathcal{B}$ may not issue the challenge ciphertext during the challenger phase. The adversary $\mathcal{A}$ returns a forged private-key in polynomial time during the guess phase.
– In a game for unauthorized access attack, by which user can exceed its authority, we hold the above game.[5]

We denote by $\mathrm{Adv}_{\mathcal{E},\mathcal{A}}(t,n)$ the advantage of adversary $\mathcal{A}$ in winning the game:

$$\begin{aligned}\mathrm{Adv}_{\mathcal{E},\mathcal{A}}(t,n) &= \frac{1}{2}\left|\Pr[\mathcal{A}_{\mathcal{E}}(\mathcal{C}_{\mathrm{i}})=\mathrm{b}]-\Pr[\mathcal{A}_{\mathcal{E}}(\mathcal{C}_{\mathrm{i}})\neq \mathrm{b}]\right| \\ &= \left|\Pr[\mathcal{A}_{\mathcal{E}}(\mathrm{C}_{\mathrm{i}})=\mathrm{b}]-\frac{1}{2}\right|\end{aligned}$$

We say that a PHE is (t,n)-secure if for all setup parameter P and all probabilistic polynomial-time adversaries $\mathcal{A}$, the function $\mathrm{Adv}_{\mathcal{E},\mathcal{A}}(t,n)$ is a negligible function of s.

Semantic security is a widely-used security notion in a public-key encryption scheme. Informally, it requires that it is infeasible to learn anything about the plaintext from the ciphertext. This security requirement is also fit for PHE scheme. We show that our encryption scheme is semantically secure agaist chosen plaintext attack (IND-CPA) under the Decision Diffie-Hellman (DDH) assumption as the following theorem:

Theorem 1. *The proposed (t,n)-PHE scheme is semantically secure under chosen plaintext attacks assuming the difficulty of Decisional Diffie-Hellman (DDH) problem in* G_q.

Obviously, semantic security is not enough to satisfy the security requirement of "1:n" encryption scheme. It is important to consider all types of potential attacks when we attempt to design the key hierarchy and broadcast scheme. The security of key hierarchy must assure that the adversary cannot gain any advantage by analyzing public-keys, ciphertexts, and user's private keys. There exist two strategies to attack the PHE scheme:

1. Privilege Attack: it focuses on changing the privileges of the granted users or getting the keys of the other users. This attack also involves two ways:
   – Collusion attack for corrupted classes, in which the corrupted users in $R=\{u_{i_k,j_k}\}_{k=1}^{t}$ wish to forge a (new or unused) key in $\{c_{i_1},\cdots,c_{i_t}\}$ (called as the corrupted classes). The aim of this attack is to avoid tracing and frame the innocent users.
   – Collusion attack for honest classes, in which the corrupted users in $R=\{u_{i_k,j_k}\}_{k=1}^{t}$ wish to forge a (new or unused) key in $C\setminus\{c_{i_1},\cdots,c_{i_t}\}$. The aim of this attack is to change the privileges in partial order hierarchy.

[5] This game may be more strict than the other two games.

2. Access Attack: it focuses on gaining the advantage of adversary to break the cryptosystem or extending the range of access by the collusion of corrupted users, especially gaining the advantage to break the revocation-based algorithm.

We would like to adopt appropriate technologies to prevent the above attacks, but the collusion attack is unavoidable in the way of technology because the traitor has been a granted user before s/he is not found. Thus traitor tracing is an efficient method to frighten the collusion attack. However, we must ensure that the traitors cannot forge an 'unused' key to avoid tracing but leave some 'foregone' clue of evidence to find them. We present such a definition for Secure Key Hierarchy (SKH) as follows:

Definition 1 (Secure Key hierarchy). *A (t,n)-PHE scheme $(\mathcal{S},\mathcal{J},\mathcal{E},\mathcal{D})$ is said to have a secure key hierarchy $\langle C,E,K\rangle$ satisfying the following conditions:*

1. Validity: *for any member $u_{i,j}$ in $c_i \in C$, the session key ek can be efficiently computed from B_l and $sk_{i,j}$, where $c_i \prec c_l$. Then for every pair $(pk_l, sk_{i,j})$ in the range of $\mathcal{G}(1^n)$ and every sequence M_n, $|M_n| \leq poly(n)$,*

$$\Pr\left[\mathcal{D}(\mathrm{sk_{i,j}}, \mathcal{E}(\mathrm{pk_l}, \mathrm{M_n})) = \mathrm{M_n}\right] \geq 1 - \frac{1}{|\mathrm{p(n)}|}; \tag{7}$$

   $\frac{1}{|p(n)|}$ *denotes negligible or negligibly small, which means that the absolute value is asymptotically smaller than any polynomial bound.*
2. Privilege attack: *for any set $R \subseteq \{u_{i_1,j_1}, \cdots, u_{i_m,j_m}\}$, $|R| \leq t$, it is computationally infeasible to compute $sk_{i,j}$ of a user $u_{i,j} \notin R$ and the (public) encryption key pk. Then for every probabilistic polynomial-time algorithm $\mathcal{A}$, every polynomial $p(\cdot)$, and all sufficiently large n,*

$$\Pr\left[\begin{array}{c}\mathcal{A}(pk, \{sk_{i_l,j_l}\}_{u_{i_l,j_l}\in R}) = sk_{i,j} \\ : sk_{i_l,j_l} \notin \{sk_{i,j}\}_{u_{i_l,j_l}\in R}\end{array}\right] < \frac{1}{|\mathrm{p(n)}|}; \tag{8}$$

   where, $pk = P \cup \{pk_i\}_{c_i \in C}$.
3. Access attack*: for any set $R \subseteq \{u_{i_1,j_1}, \cdots, u_{i_m,j_m}\}$ $|R| \leq t$, it is computationally infeasible to gain the advantage to break the revocation-based algorithm from the collusion set R and any ciphertexts $\mathcal{C}_l = \mathcal{E}^R_{pk_l}(M_n)$, where $\mathcal{E}^R$ denotes revocation-based algorithm on R and M_n is a sequence with $|M_n| \leq poly(n)$. Then for every probabilistic polynomial-time algorithm $\mathcal{A}$, every pair of polynomially-bounded functions $f, h : \{0,1\}^* \to \{0,1\}^*$ (see [17]), every polynomial $p(\cdot)$, and all sufficiently large n,*

$$\begin{aligned}&\Pr\left[\mathrm{A}\left(\begin{array}{c}pk, h(X_n), \mathcal{E}^R_{pk_l}(X_n), \\ \{sk_{i,j}\}_{u_{i,j}\in R}\end{array}\right) = \mathrm{f(X_n)}\right] \\ &< \Pr\left[\mathrm{A}\left(\begin{array}{c}pk, h(X_n), \\ \{sk_{i,j}\}_{u_{i,j}\in R}\end{array}\right) = \mathrm{f(X_n)}\right] + \frac{1}{|\mathrm{p(n)}|}.\end{aligned} \tag{9}$$

   Where, $f(X_n)$ denotes the information that the adversary tries to obtain from the plaintext X_n and $h(X_n)$ denotes a priori partial information about the plaintext.

In this definition, the condition 3) aims at the risk of revocation-based mechanism and puts forward this security requirement (tighter than Theorem 1), which conforms to the definition of 'semantic security' besides the additional key information $\{sk_{i,j}\}_{u_{i,j}\in R}$ for a set of revoked users R. As is well known, the encryption scheme is semantically secure if and only if it has indistinguishable encryptions (see Theorem 5.2.5 in [17]). So, we replace Eq. (9) with the following equation

$$\left| \begin{matrix} \Pr[\mathcal{A}(\mathrm{pk}, \{\mathrm{sk_{i,j}}\}_{\mathrm{u_{i,j}}\in \mathrm{R}}, \mathcal{E}^{\mathrm{R}}_{\mathrm{pk_l}}(\mathrm{X_n})) = 1] - \\ \Pr[\mathcal{A}(\mathrm{pk}, \{\mathrm{sk_{i,j}}\}_{\mathrm{u_{i,j}}\in \mathrm{R}}, \mathcal{E}^{\mathrm{R}}_{\mathrm{pk_l}}(\mathrm{Y_n})) = 1] \end{matrix} \right| < \frac{1}{|p(n)|}, \tag{10}$$

such that it is easier than ever to prove the security of scheme against access attack. According to this definition, we can prove the following theorem.

Theorem 2. *The proposed (t, n)-PHE scheme has a secure key hierarchy satisfying Definition 1 against the privilege attack and the access attack.*

In the proof of this theorem, the security against privilege attack includes two cases: privilege attack for honest classes and one for corrupted classes. The proofs of the above-mentioned theorems were omitted due to space limitations.[6]

6 Related Work

For a large-scale group-oriented communication, broadcast encryption was first considered [18] in 1991 and, subsequently, formally defined by Fiat and Naor [19] in 1994. Since then, it has become one attractive topic in cryptography community. In symmetric-key setting, only trusted system designer can broadcast data to the receivers. However, the public-key scheme, first introduced by Boneh *et al.* in 1999 [20], can publish a short public key, which enables anybody to broadcast data, thus overcome the deficiency of symmetric-key setting. Also, Boneh *et al.* have done massive work in the development of group-oriented encryption, e.g., Boneh, Sahai, and Waters [21] propose a fully collusion resistant traitor tracing with ciphertexts of size $O(\sqrt{n})$ and private keys of size $O(1)$ in 2006, where n is the total number of users. However, these work did not take into account the hierarchy structure.

Boneh and Franklin proposed the first fully identity-based encryption (IBE) [22] in 2001, in which the public key can be an arbitrary string such as an email address. Unfortunately, IBE does not support broadcast function unless some members can share the same private-key when they hold the same identity. According to this idea, Boneh *et al.* provided a hierarchical identity-based encryption (HIBE) system to support an organizational hierarchy [23], but this kind of hierarchy must be a tree structure and cannot provide identity-based revocation and tracing due to the global sharing of hierarchical identity/privacy-key for all users. In addition, attribute-based encryption (ABE) is also considered

[6] The interesting readers may read the full proofs in the website: crypto.ustb.edu.cn.

as an effective group communication method [24], but the existing ABE schemes have not yet been able to support the hierarchical structure.

For cryptosystems on the partial order relation, Akl and Taylor put forward a simple scheme to solve multilevel security problem in 1982. In 2005, Kim [25] proposed a new key management system for multilevel security using various one-way functions. In 2008, Chung [26] proposed a method based on the elliptic curve cryptosystem and one-way hash function to solve dynamic access problems. Another related field is *hierarchical key management with time control*. For example, in 2002, Tzeng proposed a time-bound scheme based on Lucas function [27], but it is insecure against collusion attacks by Yi and Ye. Another similar schemes based on the tamper-resistant device and the hash function were proposed by Chien [28] in 2004 and Bertino *et al.* [29] in 2008. Although these work support real-time broadcast with time control rather than common access control and digital forensics, their hierarchy techniques are worth learning for hierarchy managements. In 2007, Santis *et al.* summarized and provided several provably-secure hierarchical key assignment schemes based on an existing schemes [30].

7 Conclusion and Future Work

In this paper we construct an effective RBAC-compatible cryptosystem for cloud data encryption. In our future work, we are planning to introduce a comprehensive role-based cryptosystem to support various secure mechanisms, such as encryption, signature, and authentication. Also, we would investigate a more efficient cryptosystem to realize massive-scale conditional access control systems for the practical RBAC applications of large-scale organizations.

Acknowledgments. The authors are indebted to anonymous reviewers for their valuable suggestions. This work is supported by the National 973 Program (Grant No. 2013CB329605) and National Natural Science Foundation of China (Grant Nos. 61170264 and 61472032).

References

1. F.R. Institute: Personal data in the cloud: a global survey of consumer attitudes (2010). http://www.fujitsu.com/downloads/SOL/fai/reports/fujitsu/personal-data-in-the-cloud.pdf
2. Boneh, D., Franklin, M.: Identity-based encryption from the weil pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (2001)
3. Sahai, A., Waters, B.: Fuzzy identity-based encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (2005)
4. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: ACM Conference on CCS, pp. 89–98 (2006)

5. Ostrovsky, R., Sahai, A., Waters, B.: Attribute-based encryption with non-monotonic access structures. In: ACM Conference on Computer and Communications Security, pp. 195–203 (2007)
6. Nishide, T., Yoneyama, K., Ohta, K.: Attribute-based encryption with partially hidden ciphertext policies. IEICE Trans. **92**–**A**(1), 22–32 (2009)
7. Zhu, Y., Ahn, G.-J., Hu, H., Ma, D., Wang, S.: Role-based cryptosystem: a new cryptographic rbac system based on role-key hierarchy. IEEE Trans. Inf. Forensics Secur. **8**(12), 2138–2153 (2013)
8. Atallah, M.J., Blanton, M., Fazio, N., Frikken, K.B.: Dynamic and efficient key management for access hierarchies. ACM Trans. Inf. Syst. Secur. **12**(3), 1–43 (2009)
9. Blanton, M., Frikken, K.B.: Efficient multi-dimensional key management in broadcast services. In: Gritzalis, D., Preneel, B., Theoharidou, M. (eds.) ESORICS 2010. LNCS, vol. 6345, pp. 424–440. Springer, Heidelberg (2010)
10. Boneh, D., Boyen, X., Goh, E.-J.: Hierarchical identity based encryption with constant size ciphertext. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 440–456. Springer, Heidelberg (2005)
11. Zhu, Y., Ahn, G.-J., Hu, H., Yau, S.S., An, H.G., Hu, C.-J.: Dynamic audit services for outsourced storages in clouds. IEEE Trans. Serv. Comput. **6**(2), 227–238 (2013)
12. Wallner, D.M., Harder, E.G., Agee, R.C.: Key management for multicast: Issues and architecture. In: Internet draft draft-waller-key-arch-01.txt (1998)
13. Asano, T.: Reducing receiver's storage in CS, SD and LSD broadcast encryption schemes. IEICE Trans. Fundam. Electron. Commun. Comput. Sci. **88**(1), 203–210 (2005)
14. Halevy, D., Shamir, A.: The LSD broadcast encryption scheme. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 47–60. Springer, Heidelberg (2002)
15. Naor, D., Naor, M., Lotspiech, J.: Revocation and tracing schemes for stateless receivers. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 41–62. Springer, Heidelberg (2001)
16. Tzeng, W.-G., Tzeng, Z.-J.: A public-key traitor tracing scheme with revocation using dynamic shares. In: Public Key Cryptography, pp. 207–224 (2001)
17. Goldreich, O.: Foundations of Cryptography. Basic Application, vol. II. Cambridge University Press, New York (2004)
18. Berkovits, S.: How to broadcast a secret. In: Davies, D.W. (ed.) EUROCRYPT 1991. LNCS, vol. 547, pp. 535–541. Springer, Heidelberg (1991)
19. Fiat, A., Naor, M.: Broadcast encryption. In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS, vol. 773, pp. 480–491. Springer, Heidelberg (1994)
20. Boneh, D., Franklin, M.K.: An efficient public key traitor scheme (extended abstract). In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 338–353. Springer, Heidelberg (1999)
21. Boneh, D., Sahai, A., Waters, B.: Fully collusion resistant traitor tracing with short ciphertexts and private keys. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 573–592. Springer, Heidelberg (2006)
22. Boneh, D., Franklin, M.: Identity-based encryption from the weil pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (2001)
23. Boneh, D., Boyen, X., Goh, E.-J.: Hierarchical identity based encryption with constant size ciphertext. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 440–456. Springer, Heidelberg (2005)
24. Bethencourt, J., Sahai, A., Waters, B.: Ciphertext-policy attribute-based encryption. In: IEEE Symposium on Security and Privacy, pp. 321–334 (2007)

25. Kim, H.K., Park, B., Ha, J.C., Lee, B., Park, D.G.: New key management systems for multilevel security. In: Gervasi, O., Gavrilova, M.L., Kumar, V., Laganá, A., Lee, H.P., Mun, Y., Taniar, D., Tan, C.J.K. (eds.) ICCSA 2005. LNCS, vol. 3481, pp. 245–253. Springer, Heidelberg (2005)
26. Chung, Y.F., Lee, H.H., Lai, F., Chen, T.S.: Access control in user hierarchy based on elliptic curve cryptosystem. Inf. Sci. **178**, 230–243 (2008)
27. Tzeng, W.G.: A time-bound cryptographic key assignment scheme for access control in a hierarchy. IEEE Trans. Knowl. Data Eng. **14**(1), 182–188 (2002)
28. Chien, H.Y.: Efficient time-bound hierarchical key assignment scheme. IEEE Trans. Knowl. Data Eng. **16**(10), 1301–1304 (2004)
29. Bertino, E., Bettini, C., Ferrari, E., Samarati, P.: An access control model supporting periodicity constraints and temporal reasoning. ACM Trans. Database Syst. **23**(3), 231–285 (1998)
30. De Santis, A., Ferrara, A.L., Masucci, B.: Efficient provably-secure hierarchical key assignment schemes. In: Kučera, L., Kučera, A. (eds.) MFCS 2007. LNCS, vol. 4708, pp. 371–382. Springer, Heidelberg (2007)