New Ciphertext-Policy Attribute-Based Access Control with Efficient Revocation

Xingxing Xie, Hua Ma, Jin Li, Xiaofeng Chen

HAL Id: hal-01480196
https://hal.inria.fr/hal-01480196

New Ciphertext-Policy Attribute-Based Access Control with Efficient Revocation[*]

Xingxing Xie[1], Hua Ma[1], Jin Li[2], and Xiaofeng Chen[3]

[1] Department of Mathematics,
Xidian University, Xi'an 710071, P.R.China
`xiexingxing11@163.com, mahua@126.com`
[2] Department of Computer Science,
Guangzhou University, Guangzhou 510006, P.R.China
`jinli71@gmail.com`
[3] State Key Laboratory of Integrated Service Networks (ISN),
Xidian University, Xi'an 710071, P.R.China
`xfchen@xidian.edu.cn`

Abstract. Attribute-Based Encryption (ABE) is one of the promising cryptographic primitives for fine-grained access control of shared outsourced data in cloud computing. However, before ABE can be deployed in data outsourcing systems, it has to provide efficient enforcement of authorization policies and policy updates. However, in order to tackle this issue, efficient and secure attribute and user revocation should be supported in original ABE scheme, which is still a challenge in existing work. In this paper, we propose a new ciphertext-policy ABE (CP-ABE) construction with efficient attribute and user revocation. Besides, an efficient access control mechanism is given based on the CP-ABE construction with an outsourcing computation service provider.

Key words: Attribute-based encryption, revocation, outsourcing, re-encryption

1 Introduction

As a relatively new encryption technology, attribute-based encryption(ABE) has attracted lots of attention because ABE enables efficient one-to-many broadcast encryption and fine-grained access control system. Access control is one of the most common and versatile mechanisms used for information systems security enforcement. An access control model formally describes how to decide whether an access request should be permitted or repudiated. Particularly, in the outsourcing environment, designing an access control will introduce many challenges.

However, the user and attribute revocation is still a challenge in existing ABE schemes. Many schemes [1, 3, 9] are proposed to cope with attribute-based

---

[*] Corresponding author: Xiaofeng Chen (xfchen@xidian.edu.cn)

access control with efficient revocation. The most remarkable is the scheme proposed by J.Hur and D.K.Noh, which realizes attribute-based access control with efficient fine-grained revocation in outsourcing. However, in the phase of key update, the data service manager will perform heavy computation at every time of update, which could be a bottleneck for the data service manager. Moreover, in the outsourcing environment, external service provider [14, 15] is indispensable. Thus, in this paper, we attempt to solve the problem of efficient revocation in attribute-based data access control using CP-ABE for outsourced data.

1.1 Related Work

For the ABE, key-policy ABE (KP-ABE) and ciphertext-policy ABE (CP-ABE) are more prevalent than the others. To take control of users' access right by a data owner, we specify CP-ABE as the data outsourcing architecture.

Attribute Revocation Recently, several attribute revocable ABE schemes have been announced [2–4]. Undoubtedly, these approaches have two main problems, which consists of security degradation in terms of the backward and forward security [1, 5]. In the previous schemes, the key authority periodically announce a key update, that will lead to a bottleneck for the key authority. Two CP-ABE schemes with immediate attribute revocation with the help of semi-honest service provider are proposed in [6, 7]. However, achieving fine-grained user access control failed. Junbeom et al. [1] proposed a CP-ABE scheme with fine-grained attribute revocation with the help of the honest-but-curious proxy deployed in the data service provider.

User Revocation In [8], a fine-grained user-level revocation is proposed using ABE that supports negative clause. In the previous schemes [8, 9], a user loses all the access rights to the data when he is revoked from a single attribute group. Attrapadung and Imai [10] suggested another user-revocable ABE schemes, in which the data owner should take full control of all the membership lists that leads to be not applied in the outsourcing environments.

1.2 Our Contribution

In this study, aiming at reducing the overhead computation at data service manager, we propose an ciphertext policy attribute-based access control with efficient revocation. This construction is based on a CP-ABE construction with efficient user and attribute revocation. Compared with [1], in our proposed construction, in the phase of key update, the computation operated by the data service manager will reduce by half. In Table 1 we summarize the comparisons between our proposed scheme and [1] in terms of the computations in the phase of key update. Furthermore, we formally show the security proof based on security requirement in the access control system.

2 Systems and Models

2.1 System Description and Assumptions

There are four entities involved in our attribute-based access control system:

- Trusted authority. It is the party that is fully trusted by all entities participating in the data outsourcing system.
- Data owner. It is a client who owns data and encrypts the outsourced data.
- User. It is an entity who would like to access the cryptographic data.
- Service provider. It is an entity that provides data outsourcing service. The data servers are responsible for storing the outsourced data. Access control from outside users is executed by the data service manager, which is assumed to be honest-but-curious.

2.2 Threat Model and Security Requirements

- Data confidentiality. It is not allowed to access the plaintext if a user's attributes do not satisfy the access policy. In addition, unauthorized data service manager should be prevented from accessing the plaintext of the encrypted data that it stores.
- Collusion-resistance. Even if multiple users collaborate, they are unable to decrypt encrypted data by combining their attribute keys.
- Backward and forward secrecy. In our context, backward secrecy means that if a new key is distributed for the group when a new member joins, he is not able to decrypt previous messages before the new member holds the attribute. On the other hand, forward secrecy means that a revoked or expelled group member will not be able to continue accessing the plaintext of the subsequent data exchanged (if it keeps receiving the messages), when the other valid attributes that he holds do not satisfy the access policy.

3 Preliminaries and Definitions

3.1 Bilinear Pairings

Let $\mathbb{G}$ and $\mathbb{G}_T$ be two cyclic group of prime order p. The Discrete Logarithm Problem on both $\mathbb{G}$ and $\mathbb{G}_T$ are hard. A bilinear map e is a map function $e: \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ with the following properties:

1. Bilinearity: For all $A, B \in \mathbb{G}$, and $a, b \in \mathbb{Z}_p^*$, $e(A^a, B^b) = e(A, B)^{ab}$.
2. Non-degeneracy: $e(g, g) \neq 1$, where g is the generator of $\mathbb{G}$.
3. Computability: There exsits an efficient algorithm to compute the pairing.

3.2 Decisional Bilinear Diffie-Hellman Exponent (BDHE) Assumption [12]

The decisional BDHE problem is to compute $e(g,g)^{a^{q+1}s} \in \mathbb{G}_T$, given a generator g of $\mathbb{G}$ and elements $\overrightarrow{y} = (g_1, \cdots, g_q, g_{q+1}, \cdots, g_{2q}, g^s)$ for $a, s \in \mathbb{Z}_p^*$. Let g_i denote g^{a^i}.

An algorithm $\mathcal{A}$ has advantage $\epsilon(\kappa)$ in solving the decisional BDHE problem for a bilinear map group $\langle p, \mathbb{G}, \mathbb{G}_{\mathbb{T}}, e \rangle$, where κ is the security parameter, if $|Pr[\mathcal{B}(\overrightarrow{y}, g, T = e(g,g)^{a^{q+1}s}) = 0] - Pr[\mathcal{B}(\overrightarrow{y}, T = R) = 0] | \geq \epsilon(\kappa)$.

$\langle p, \mathbb{G}, \mathbb{G}_{\mathbb{T}}, e \rangle$ is deemed to satisfy the decisional BDHE assumption. when for every polynomial-time algorithm (in the security parameter κ) to solve the decisional BDHE problem on $\langle p, \mathbb{G}, \mathbb{G}_{\mathbb{T}}, e \rangle$, the advantage $\epsilon(\kappa)$ is a negligible function.

3.3 System Definition and Our Basic Construction

Let $\mathcal{U} = \{u_1, \cdots, u_n\}$ be the whole of users. Let $\mathcal{L} = \{1, \cdots, p\}$ be the universe of attributes that defines, classifies the user in the system. Let $G_i \subset \mathcal{U}$ be a set of users that hold the attribute i. Let $\mathcal{G} = \{G_1, \cdots, G_p\}$ be the whole of such attribute groups. Let K_i be the attribute group key that is possessed by users, who own the attribute i.

Ciphertext Policy Attribute-Based Access Control with User Revocation.

Definition 1. *A CP-ABE with user revocation capability scheme consists of six algorithms:*

- **Setup:** *Taking a security parameter k, this algorithm outputs a public key PK and a master secret key MK.*
- **KeyGen**(MK, S, U)**:** *Taking the MK, and a set of attributes $S \subseteq \mathcal{L}$ and users $U \subseteq \mathcal{U}$, this algorithm outputs a set of private attributes keys SK for each user.*
- **KEKGen**(U)**:** *Taking a set of users $U \subseteq \mathcal{U}$, this algorithm outputs KEKs for each user, which will be used to encrypt attribute group keys.*
- **Encrypt**(PK, M, $\mathcal{T}$)**:** *Taking the PK, a message M and an access structure $\mathcal{T}$, this algorithm outputs the ciphertext CT .*
- **Re-Encrypt**(CT, G)**:** *Taking ciphertext CT and attributes groups G, this algorithm outputs re-encrypted CT'.*
- **Decrypt**(CT', SK, K_S)**:** *The decryption algorithm takes as input the ciphertext $CT^{'}$, a private key SK, and a set of attribute group keys K_S. The decryption can be done.*

4 Ciphertext Policy Attribute-Based Access Control with Efficient Revocation

4.1 KEK Construction

In our scheme, KEK tree will be used to re-encrypt the ciphertext encrypted by the owner, which is constructed by the data service manager as in Fig.1. Now, some basic properties of the KEK tree will be presented as [1].
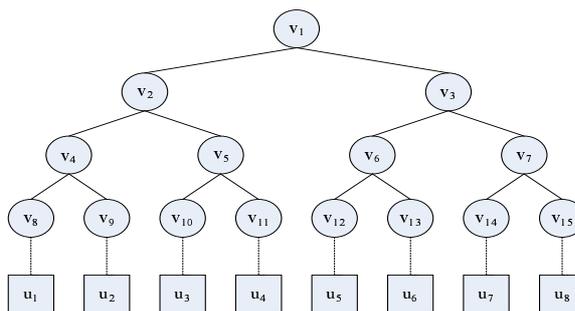

Fig. 1. KEK tree attribute group key distribution

4.2 Our Construction

Let $e : \mathbb{G} \times \mathbb{G} \longrightarrow \mathbb{G}_T$ be a bilinear map of prime order p with the generator g. A security parameter, κ, will decide the size of the groups. We will additionally employ a hash function $H : \{0,1\}^* \longrightarrow \mathbb{G}$ that we will model as a random oracle.

System Setup and Key Generation The trusted authority (TA) first runs Setup algorithm by choosing a bilinear map : $\mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ of prime order δ with a generator g. Then, TA chooses two random $\alpha, a \in Z_p$. The public parameters are $PK = \{\mathbb{G}, g, h = g^a, e(g,g)^\alpha\}$. The master key is $MK = \{g^\alpha\}$, which is only known by the TA.

After executing the Setup algorithm producing PK and MK, each user in U needs to register with the TA, who verifies the user's attributes and issues proper private keys for the user. Running $KeyGen(MK, S, U)$, the TA inputs a set of users $U \subseteq \mathcal{U}$ and attributes $S \subseteq \mathcal{L}$, and outputs a set of private key components corresponding to each attribute $j \in S$. The key generation algorithm is presented as follows:

1. Choose a random $r \in \mathbb{Z}_p^*$, which is unique to each user.
2. Compute the following secret value to the user $u \in U$ as:

$$SK = (K = g^\alpha g^{ar}, L = g^r, \forall j \in S : D_j = H(j)^r)$$

After implementing above operations, TA sends the attribute groups G_j [1] for each $j \in S$ to the data service manager.

KEK Generation After obtaining the attribute groups G_j for each $j \in S$ from the TA, the data service manager runs $KEKGen(U)$ and generates KEKs for users in U. Firstly, the data service manager sets a binary KEK tree for the unniverse of users $\mathcal{U}$ just like that described above. The KEK tree is responsible for distributing the attribute group keys to users in $U \subseteq \mathcal{U}$. For instance, u_3 stores $PK_3 = \{KEK_{10}, KEK_5, KEK_2, KEK_1\}$ as its path keys in Fig.2.

Then, in the data re-encryption phase, the data service manager will encrypt the attribute group keys by no means the path keys, i.e. KEKs. The method of the key assignment is that keys are assigned randomly and independently from each other, which is information theoretic.

Data Encryption To encrypt the data M, a data user needs to specify a policy tree $\mathcal{T}$ over the universe of attributes $\mathcal{L}$. Running $Encrypt(PK, M, \mathcal{T})$, the data M is enforced attribute-based access control. The policy tree $\mathcal{T}$ is defined as follows.

For each node x in the tree $\mathcal{T}$, the algorithm chooses a polynomial q_x, which is chosen in a top-down manner, starting from the root node R and its degree d_x is one less than the threshold value k_x of the node, that is, $d_x = k_x - 1$. For the root node R, it randomly chooses an $s \in \mathbb{Z}_p^*$ and sets $q_R(0) = s$. Except the root node R, it sets $q_x(0) = q_{p(x)}(index(x))$ and chooses d_x other points randomly to completely define q_x for any other node x. Let Y be the set of leaf nodes in $\mathcal{T}$. The ciphertext is then constructed by giving the policy tree $\mathcal{T}$ and computing

$$CT = (\mathcal{T}, \widetilde{C} = Me(g,g)^{\alpha s}, C = g^s,$$

$$\forall y \in Y : C_y = g^{aq_x(0)} \cdot H(y)^{-s})$$

After constructing CT, the data owner outsources it to the service provider securely.

Data Re-Encryption On receiving the ciphertext CT, the data service manager re-encrypts CT using a set of the membership information for each attribute group $G \subseteq \mathcal{G}$. The re-encryption algorithm progresses as follows:

1. For all $G_y \in G$, chooses a random $K_y \in \mathbb{Z}_p^*$ and re-encrypts CT as follows:

$$CT' = (\mathcal{T}, \widetilde{C} = Me(g,g)^{\alpha s}, C = g^s,$$

$$\forall y \in Y : C_y = (g^{aq_x(0)} \cdot H(y)^{-s})^{K_y})$$

2. After re-encrypting CT, the data service manager needs to employ a method to deliver the attribute group keys to valid users. The method we used is a symmetric encryption of a message M under a key K, in other words, $E_K : \{0,1\}^k \longrightarrow \{0,1\}^k$, as follow:

$$Hdr = (\forall y \in Y : \{E_K(K_y)\}_{K \in KEK(G_y)})$$

After above all operations, the data service manager responds with (Hdr, CT') to the user sending any data request.

Data Decryption Data decryption phase consists of the attribute group key decryption from Hdr and message decryption.

Attribute group key decrypt To execute data decryption, a user u_t first decrypt the attribute group key for all attributes in S that the user holds from Hdr. If the user $u_t \in G_j$, he can decrypt the attribute group key K_j from Hdr using a KEK that is possessed by the user. For example, if $G_j = \{u_1, u_2, u_5, u_6\}$ in Fig.2, u_5 can decrypt the K_j using the path key $KEK_6 \in PK_5$. Next, u_t updates its secret key as follows:

$$SK = (K = g^\alpha g^{ar}, \forall j \in S : D_j = H(j)^r, L = (g^r)^{1/K_j})$$

Message decrypt Once the user updates its secret key, he then runs the $Decrypt(CT', SK, K_S)$ algorithm as follows. The user runs a recursive function $DecryptNode(CT', SK, R)$, R is the root of $\mathcal{T}$. The recursion function is the same as defined in [2]. And if x is a leaf node, then $DecryptNode(CT', SK, x)$ is proceeded as follow when $x \in S$ and $u_t \in G_x$:

$$
\begin{aligned}
Decrypt(CT', SK, x) &= e(C_x, L) \cdot e(C, D_x) \\
&= e((H(x)^{-s} \cdot g^{aq_x(0)})^{K_x}, (g^r)^{1/K_x}) \cdot e(g^s, H(x)^r) \\
&= e(g, g)^{raq_x(0)}
\end{aligned}
$$

Now we consider the recursion when x is a nonleaf node processed as follows: $\forall z$ is the child of x, it calls $DecryptNode(CT', SK, z)$ and stores the output as F_z. Let S_x be an arbitrary k_x-sized set of child nodes z, then computes:

$$
\begin{aligned}
F_x &= \prod_{z \in S_x} F_z^{\Delta_{i, S_x'}(0)}, where_{S_x' = \{index(z) : z \in S_x\}}^{i = index(x),} \\
&= \prod_{z \in S_x} (e(g, g)^{r \cdot aq_z(0)})^{\Delta_{i, S_x'}(0)} \\
&= \prod_{z \in S_x} (e(g, g)^{r \cdot aq_{p(z)}(index(z))})^{\Delta_{i, S_x'}(0)} \\
&= \prod_{z \in S_x} (e(g, g)^{r \cdot aq_x(i)})^{\Delta_{i, S_x'}(0)} \\
&= e(g, g)^{r \cdot aq_x(0)}
\end{aligned}
$$

Where $i = index(z)$ and $S_x' = \{index(z) : z \in S_x\}$. Finally, if x is the root node R of the access tree $\mathcal{T}$, the recursive algorithm returns $A = DecryptNode(CT', SK, R) = e(g, g)^{ras}$. And the algorithm decrypts the ciphertext by computing

$$\widetilde{C}/(e(C, K)/A) = \widetilde{C}/(e(g^s, g^\alpha g^{ra})/e(g, g)^{ras}) = M.$$

5   Key Update

In this section, when a user wants to leave or join several attribute groups, he needs to send the attributes changed to TA. Without loss of generality, assume

there is any membership change in G_j, which is equal to that a user comes to hold or drop an attribute j at the some instance. Next, we progress the update procedure as follows:

1. The data service manager selects a random $s' \in \mathbb{Z}_p^*$ and a K_i', and re-encrypts the ciphertext CT' using PK as

$$
\begin{aligned}
CT^{'} &= (\mathcal{T}, \widetilde{C} = Me(g,g)^{\alpha(s+s')}, C = g^{(s+s')}, \\
C_i &= (g^{a(q_x(0)+s')} \cdot H(i)^{-s})^{K_i'} \\
&\quad \forall y \in Y \setminus \{i\} : C_y = (g^{a(q_x(0)+s')} \cdot H(y)^{-s})^{K_y}).
\end{aligned}
$$

2. After updating the ciphertext, the data service manager selects new minimum cover sets for G_i changed and generates a new header message as follows:

$$
\begin{aligned}
Hdr &= (\{E_K(K_i')\}_{K \in KEK(G_i)}, \\
&\quad \forall y \in Y \setminus \{i\} : \{E_K(K_y)\}_{K \in KEK(G_y)}).
\end{aligned}
$$

6 Efficiency Analysis

In this section, we analyze the efficiency of the proposed schemes with the scheme [1]. Table 1 shows the comparisons between our scheme and scheme [1] in terms of the computations in the phase of key update. In our scheme, the number of exponentiations is reduced to $\omega + 3$. However, in the scheme [1], the number of exponentiations unexpectedly is $2\omega + 3$. Thus, it will enormously improve computational efficiency.

Table 1. Result Comparison

	the Number of Exponentiations of Key Update
Scheme one	$2\omega + 3$
Our scheme	$\omega + 3$

7 Security

In this section, the security of the proposed scheme is given based on the security requirements discussed in Section 2.

Theorem 1. *Collusion Resistance. The proposed scheme is secure to resist collusion attack.*

Proof. In CP-ABE, the secret s sharing is embedded into a ciphertext, and to decrypt a ciphertext, a user or a colluding attacker needs to recover $e(g,g)^{\alpha s}$. To recover $e(g,g)^{\alpha s}$, the attacker must pair C_x from the ciphertext and D_x from the other colluding users private key for an attribute x. However, every user's private key is uniquely generated by a random r. Thus, even if the colluding users are all valid, the attacker can not recover $e(g,g)^{\alpha s}$.

Theorem 2. *Data Confidentiality.* *The proposed scheme prevents unauthorized users and the curious service provider from acquiring the privacy of the outsourced data.*

Proof. Firstly, if the attributes held by a user don't satisfy the tree policy $\mathcal{T}$, the user will not recover the value $e(g,g)^{ras}$, which leads the ciphertext not to be deciphered. Secondly, when a user is revoked from some attribute groups that satisfy the access policy, he will lose the updated attribute group key. If the user would like to decrypt a node x for corresponding attribute, he needs to pair C_x from the ciphertext and L encrypted by K_x from its private key. As the user cannot obtain the updated attribute group key K_x, he cannot decrypt the value $e(g,g)^{raq_x(0)}$. Ultimately, Since we assume that the service provider is honest-but-curious, the service provider cannot be totally trusted by users. The service provider is available to the ciphertext and each attribute group key. However, any of the private keys for the set of attributes are not given to the data service manager. Thus, the service provider will not decrypt the ciphertext.

Theorem 3. *Backward and Forward Secrecy.* *For backward and forward secrecy of the outsourced data, the proposed scheme is secure against any newly joining and revoked users, respectively.*

Proof. When a user comes to join some attribute groups, the corresponding attribute group keys are updated and delivered to the user securely. Even if the user has stored the previous ciphertext exchanged and the holding attributes satisfy the access policy, he cannot decrypt the pervious ciphertext. That is because, though he could succeed in computing $e(g,g)^{ra(s+s')}$, it will not help to recover the value $e(g,g)^{\alpha s}$ from the updated ciphertext.

Furthermore, when a user comes to leave some attribute groups, the corresponding attribute group keys are updated and not delivered to the user. As the user cannot obtain the updated attribute group keys, he cannot decrypt any nodes corresponding to the updated attributes. Moreover, even if the user has stored $e(g,g)^{\alpha s}$, he cannot decrypt the subsequent value $e(g,g)^{\alpha(s+s')}$. Because he is not available to random s'.

8 Conclusion

In this paper, aiming at improving the efficiency of revocation to make CP-ABE widely deployed in access control, we introduced a new CP-ABE construction. In this construction, the overall computation of key update become less. Furthermore, the security proof is also shown based on access control security requirements.

10

Acknowledgements

References

1. J. Hur and D. K. Noh. Attribute-based Access Control with Efficient Revocation in Data Outsourcing System. *IEEE Transactions on Parallel and Distributed System,* pages 1214–1221, 2011.
2. J. Bethencourt, A. Sahai, and B. Waters. Ciphertext-Policy Attribute-Based Encryption. *Proc. IEEE Symp. Security and Privacy,* pages 321–334, 2007.
3. A. Boldyreva, V. Goyal, and V. Kumar. Identity-Based Encryption with Efficient Revocation. *Proc. ACM Conf. Computer and Comm. Security,* pages 417–426, 2008.
4. M. Pirretti, P. Traynor, P. McDaniel, and B. Waters. Secure Attribute-Based Systems. *Proc. ACM Conf. Computer and Comm. Security,* 2006.
5. S. Rafaeli and D. Hutchison. A Survey of Key Management for Secure Group Communication. *ACM Computing Surveys,* 35(3):309–329, 2003.
6. L. Ibraimi, M. Petkovic, S. Nikova, P. Hartel, and W. Jonker. Mediated Ciphertext-Policy Attribute-Based Encryption and Its Application. *Proc. Int'l Workshop Information Security Architecture (CSAW'07),* 2007.
7. S. Yu, C. Wang, K. Ren, and W. Lou. Attribute Based Data Sharing with Attribute Revocation. *Proc. ACM Symp. Information, Computer and Comm. Security (ASIACCS'10),* 2010.
8. R. Ostrovsky, A. Sahai, and B. Waters. Attribute-Based Encryption with Non-Monotonoc Access Structures. *Proc. ACM Conf. Computer and Comm. Security,* pages 195–203, 2007.
9. X. Liang, R. Lu, X. Lin, and X. Shen. Ciphertext Policy Attribute Based Encryption with Efficient Revocation. *technical report, Univ. of Waterloo,* http://bbcr.uwaterloo.ca/ x27liang/papers/abe/%20with%20revocation.pdf, 2011.
10. N. Attrapadung and H. Imai. Conjunctive Broadcast and Attribute-Based Encryption. *Pairing '09:Proc.Int'l Conf.Palo Alto on Pairing-Based Cryptography,* pages 248–265, 2009.
11. Z. Zhou, and D. Huang. Efficient and Secure Data Storage Operations for Mobile Cloud Computing. *Cryptology ePrint Archive,* Report: 2011/185, 2011.
12. B. Waters. Ciphertext-Policy Attribute-Based Encryption: An Expressive, Effcient, and Provably Secure Realization. *Communications of the ACM,* pages 53–70, 2011.
13. A. Shamir. How to Share a Secret. *Computer Science,* pages 612–613, 1979.
14. J. Wang, H. Ma, Q. Tang, J. Lin, H. Zhu, S. Ma, X. Chen. A New Efficient Verifiable Fuzzy Keyword Search Scheme *Journal of wireless mobile Networks, Ubiquitous Computing and Dependable Applications,* 3(4):61–71, 2012.
15. Y. Zhao, X. Chen, H. Ma, Q. Tang, H. Zhu. A New Trapdoor-indistinguishable Public Key Encryption with Keyword Search. *Journal of wireless mobile Networks, Ubiquitous Computing and Dependable Applications,* 3(1/2):72–81, 2012.