Hong and Sun *Journal of Cloud Computing: Advances, Systems and Applications* (2016) 5:2
DOI 10.1186/s13677-016-0052-1

Journal of Cloud Computing
a SpringerOpen Journal

RESEARCH Open Access

An efficient and traceable KP-ABS scheme with untrusted attribute authority in cloud computing

Hanshu Hong and Zhixin Sun*

Abstract

ABE has been widely applied for secure data protection in cloud computing. In ABE, user's private keys are generated by attribute authority, thus, attribute authority has the ultimate privileges in the system and can impersonate any users to forge valid signatures. Once the attribute authority become dishonest or be invaded in cloud systems, the system's security will be at risk. To better solve the problem mentioned above, in this paper, we propose a key-policy attribute based signature scheme with untrusted authority and traceability (KP-ABS-UT). In our scheme, the signer's private key is composed by two components: one part is distributed by attribute authority and the other part is chosen privately by the signer's self. Thus attribute authority cannot forge any signatures which should be signed by legal users. Besides, our scheme introduces an entity "tracer", which can trace the identity of signer when necessary. By security analysis and efficiency comparison, we prove our KP-ABS-UT scheme meets the requirements of unforgeability as well as lower computation cost.

Keywords: Access structure, Untrusted authority, Traceability

Introduction

With the various information resources increasing rapidly in the cloud, users are faced with urgent problems like how to make data sharing among resources efficiently and securely. On this occasion, Sahai proposed a new cryptographic primitive named "ABE" (attribute based encryption) [1]. In ABE mechanism [1–4], a user's identity is described by several attributes rather than a single string. A data receiver is capable of getting access to the data when the attributes he possesses match with the structure made by the data owner. KP-ABE [2, 3] is a typical class of ABE.As is shown in Fig. 1, a user's private key corresponds to an access tree structure. Each leaf node stands for the attribute a user owns; the non-leaf node describes the access policy of these attributes. The ciphertext corresponds to an attribute set such as {Attr1, Attr4}. Due to its capability of providing flexible access control for data sharing between users, KP-ABE is gradually becoming an effective tool for secure data sharing in complex networks.

After the proposal of ABE, ABS (attribute based signature) has been developed as a primitive to solve the data authentication problem in attribute based cryptosystem. ABS was originally proposed by Maji et al. in [5]. In ABS mechanism [6–9], a user is capable of signing a message using the private key component of the attribute set he owns. A receiver validates the signature by utilizing system public parameters. The signature can be verified to an attribute set or access structure which the signer possesses. Since the advent of the notion, many ABS schemes have been proposed. However, most of the existing ABS schemes have one thing in common, that is the attribute authority has the ultimate privileges in the whole system. In order to keep the whole system running safely, attribute authority must be honest and highly protected. In the open network systems, attribute authority are vulnerable to external as well as internal threat. Once being invaded, it can impersonate any users to forge legal signatures, which will threaten the whole security of the system. What's more, anonymity is an important feature of common attribute based signature

* Correspondence: sunzx@njupt.edu.cn
Key Lab of Broadband Wireless Communication and Sensor Network Technology, Ministry Education, Nanjing University of Posts and Telecommunications, Nanjing, China

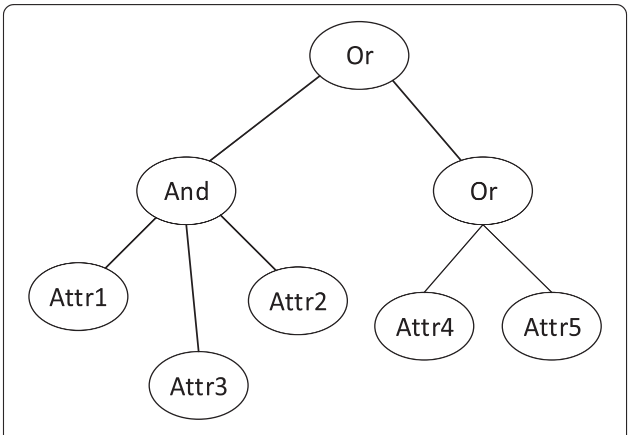

Fig. 1 Access structure for KP-ABE

mechanism. However, some malicious users may take advantage of this feature to release illegal information without taking responsibility.

To better solve the problems discussed above, in this paper, we construct a KP-ABS-UT (key-policy attribute based signature with untrusted authority and traceability) scheme, which has the following merits: (1) the signature is unforgeable, even attribute authority cannot impersonate any users to forge legal signatures; (2) the overall computation cost of the whole process of signing and verifying is reduced sharply; (3) the signer's identity can be traced by the system administer if necessary.

The rest sections are arranged as follows:

In section 2, we give the literature review and fundamental mathematical preliminaries and notions. The security model and detailed algorithms of our KP-ABS-UT scheme are constructed in section 3 and section 4. Section 5 gives the security proof and performance comparison of our scheme. At last, the conclusion of this paper and prospects on future directions are made in section 6.

Related works and preliminaries

Related works

The notion of ABS was first proposed by Maji in [5]. Since then, many ABS schemes have been proposed by researchers worldwide. Existing literatures of ABS can be classified to three types in terms of the data access structure :(1)ABS using threshold structure [10–12]; (2) ABS using LSSS access structure [13–15];(3) ABS using access tree [16], which has been illustrated in Fig. 1. Besides access structure, different ABS schemes have different advantages and performances. A. Escala et al. in [13] proposed a user-revocable attribute based signature. If a user drops some of the attributes, his signing privilege can be exactly withdrawn. Their scheme also achieves adaptively security in the standard model. S. Shahandashti et al. in [10] construct a threshold ABS and applied it to credential systems. Their scheme enables a signature holder to prove possession of signatures by revealing only the relevant attributes of the signer, hence providing signer attribute privacy for the signature holder. S. Kumar et.al in [11] proposed an ABS which is equipped with multiple threshold access structure. Their scheme is efficient as well as stretchable. Tatsuaki et al. in [16] proposed a decentralized multi-authority ABS. In their scheme, due to the introducing of multiple authorities, thus no central CA is needed. However, this also brings about other problems such as parameter synchronization, time synchronization, etc. Besides, the efficiency of their scheme can be further improved. S.L Ding et al. proposed a traceable ABS in [14], which allows PKG and the issuing authority join together to trace the identity of a malicious signer. However, since the trace algorithm needs frequent participation of PKG, it may be exposed to more internal and external security risks. Li et al. in [15] proposed a novel ABS with hidden attribute property. In their scheme, anonymous user revocation is achieved. Liu et al. in [17] proposed an attribute based multisignature scheme, which allows a number of users to participate to authenticate a message with only one signature. Their scheme is shown to be secure and is more appropriate to be applied for wireless communications. Chen et al. in [9] proposed a new paradigm named "Outsourced ABS". The computation load in their scheme is reduced sharply by delegating most of the computation work to a semi-trusted server, thus this will relieve the terminal devices from heavy computation burden. The high efficiency and security level makes their scheme an excellent method for providing data authentication in cloud computing. Xu et al. in [18] propose an ABS scheme with dynamic user revocation. Their scheme has superior performance with regard to scalability, which is able to be applied to massive data storage environments.

The above schemes have laid solid foundations for filling the gap between theoretical proposal and practical application of ABS. However, in the above schemes, the attribute authority has the ultimate privilege in the system. Even worse, it can forge as any legal users and generates a valid signature. Once the attribute authority become dishonest, the system will be exposed to potential risks. Although ABS with multiple attribute authorities has been proposed in [16], if these attribute authorities collude with each other to obtain a user's private key, a legal signature can still be successfully forged. Thus, it is of significance to cut down the privilege of the attribute authority in ABS systems.

Bilinear pairings

Denote G_1 and G_2 to be cyclic groups of prime order q. Let g be a generator of G_1. A bilinear pairing $\hat{e}: G_1 \times G_1 \rightarrow G_2$ has three features [19]:

Bilinearity: For $a, b \in Z_q$, $\hat{e}(g^a, g^b) = \hat{e}(g, g)^{ab}$.
Non-degeneracy: There exists $P, Q \in G_1$ which satisfy $\hat{e}(g, g) \neq 1$.
Computability: The value of $\hat{e}(u, v)$ can be calculated for any $u, v \in G_1$.

Hardness assumption

Decision Bilinear Diffie-Hellman problem:

Picks random numbers $a, b, c, z \in Z_q^*$, assuming that the value of (g, g^a, g^b, g^c, z) are given, no probabilistic polynomial-time algorithm can distinguish the tuples $(A = g^a, B = g^b, C = g^c, \hat{e}(g, g)^{abc})$ and $(A = g^a, B = g^b, C = g^c, \hat{e}(g, g)^z)$ with a non-negligible probability.

Lagrange Interpolation function

For a polynomial $p(x)$ in Z_q of order $d - 1$ and a set S in Z_q with the size $|S| = d$ is computed by $p(x) = \sum_{i \in S} p(i) \prod_{j \in S, j \neq i} \frac{x-j}{i-j}$.

Our model and assumptions

Formal model of our scheme

In order to clearly describe our KP-ABS-UT, we define some notations listed in Table 1.

The model of our system is shown in Fig. 2. Our model consists of 4 entities: “AA” (Attribute authority), “tracer”, “signer” and “receiver”. AA provides system’s public parameters, generates part of user’s private key and distributes them to users. After receiving the original key generated by AA, signer re-generates his own private key by adding a secret component. Then signer signs the plaintext using the private key generated by himself and uploads the signature to cloud center. Cloud center provides secure data storage and responses the access request made by data receiver. Receiver verifies if the signature is a valid one. Tracer is responsible for revealing the signer’s identity when necessary.

Table 1 Notations and their corresponding meanings

Notations	Meanings
G_1, G_2	Groups
$\hat{e}$	Bilinear paring operation
H_1	Hash function
PK	System public parameters
MK	System master key
AA	Attribute authority
U	Global attribute set
A_i	Attribute i
D_i	Private key of attribute i
M	Plaintext
T_i	Access structure
x	Node in the access tree
$\Delta_{i,S(x)}$	Lagrange interpolation function
q_x	Polynomial defined by AA
p_x	Polynomial defined by signer
v	Signature to be verified
id_l	Identity of user l

Formulized definitions of algorithms in KP-ABS-UT:

Setup$\{1^\lambda\} \rightarrow \{PK, MK\}$:This algorithm is operated by AA. Take in a security parameter, it generates PK and MK. PK is shared by users while MK is kept private by AA.
Private key generation$\{PK, MK, T_k, s\} \rightarrow \{D_{i,T_k,id_l}\}$: The private key generation algorithm is an interaction between AA and a user with an access structure T_k. On input PK, MK, T_k, AA firstly returns D_{i,T_k} to be the initial attribute private key for T_k. Then each signer embeds a secret component into D_{i,T_k} to re-generate his own private key D_{i,T_k,id_l}.Signer sends certain information referring to the secret component to tracer. Tracer assigns a unique identity number to the signer and stores the relationship of each signer with his identity.
Sign$\{D_{i,T_k,id_l}, PK, M, \sigma_{id_l}\} \rightarrow \{v\}$: This algorithm is run by a signer which the systems public parameter PK, a plaintext M, signer’s private key D_{i,T_k,id_l} and a mathematical constant σ_{id_l} which can reveal signer’s identity as input. Then the algorithm outputs a signature v for the plaintext M.
Verify$\{PK, v, M\} \rightarrow \{\theta\}$: This algorithm is run by the receiver. On input PK and the plaintext M with the signature, it outputs a value θ. If $\theta = 1$ then the signature is a valid one, if $\theta = 0$ then the signature is invalid.
Trace$\{MK, v\} \rightarrow \{id_l\}$: This algorithm is run by the tracer. On input the signature v,the tracer can pinpoint the exact identity of a signer.

The essential unforgeability of our KP-ABS-UT

Definition 1

Our KP-ABS-UT has the existential unforgeability if no *Adversary* has non-negligible advantage in the following game played by a *Challenger* and an *Adversary*.

Setup : *Challenger* runs *Setup* procedure to obtain the system parameters and sends PK to *Adversary*.

Sign queries : *Adversary* chooses an access control structure T_i,a plaintext M. *Challenger* calculates the attribute private key D_{i,T_k,id_l} and runs the *Sign* procedure to calculate the signature $v = Sign(PK, M, D_{i,T_k,id_l})$.After then, *Challenger* sends the signature v to *Adversary*.

Challenge : *Adversary* chooses a plaintext M^*,a challenging access structure T_c , and calculates the signature v^*.

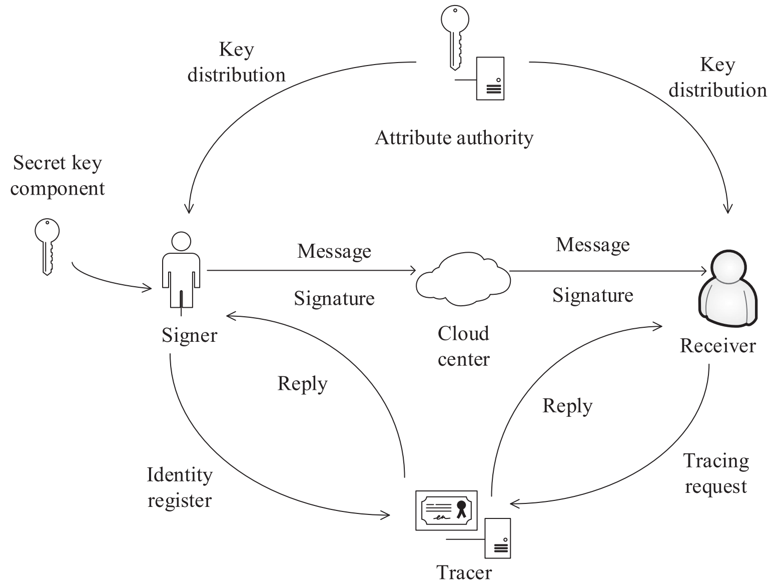

Fig. 2 The system model of KP-ABS-UT in cloud computing

Challenger verifies the signature by running the procedure *Verify*(PK, v^*) and outputs a value θ.

Adversary wins if the output of *Verify* algorithm is valid.

Denote $Adv(A) = |Pr[\theta = 1]|$ to be advantage of *Adversary*.

Our construction to KP-ABS-UT scheme

Concrete algorithms:

Setup: Let G_1 and G_2 be two cyclic groups of prime order p, while g is the generator of G_1. Let $\hat{e} : G_1 \times G_1 \rightarrow G_2$ be a bilinear paring. Define $H_1 : \{0,1\}^* \rightarrow Z_p$.AA picks $t_i \in Z_p^*$ for each A_i in the system. Besides, AA picks another secret number $y \in Z_p^*$ and calculates $\hat{e}(g,g)^y$.Set MK to be $\{t_i, y\}$ while PK to be $\{G_1, G_2 , p, g, H_1, \hat{e}(g,g)^y, g^{t_i}\}$.

Private key generation: AA randomly chooses a polynomial q_x for each node x in the access tree and sets the degree d_x of q_x to be one less than the threshold value k_x of that node ($d_x = k_x - 1$). For any other node (except root node) in the access tree, let $q_x(0) = q_{parent(x)}^{index(x)}$. For the root node AA sets $q_{root}(0) = y$. To avoid the signature forgery which can be made by AA, the signer also chooses a polynomial p_x likewise and picks a secret number s. For the root node AA sets $q_{root}(0) = y$ while the signer sets $p_{root}(0) = s$. Once the polynomials have been decided, the private key is of each leaf node x in the access tree can be denoted by $D_{i,T_k,id_l} = \left\{ g^{\frac{q_x(0).p_x(0)}{t_i}}, i \in T_k \right\}$. Signer sends the value of g^s to tracer. For each signer, tracer chooses a global unique identifier $id_l \in Z_p^*$ to describe his identity (Without loss of generality, the signer's identity is denoted by id_l). Tracer calculates $\sigma_{id_l} = (g^s)^{id_l}$ and sends σ_{id_l} back to the signer. Meanwhile, it stores a list recording the relationship between singer's unique identifier and σ_{id_l}.

Sign: For a plaintext M, the signer picks a random number $r \in Z_p^*$ and calculates:

$$C_0 = \hat{e}(g,g)^{yrs}, C_1 = \hat{e}(g,g)^{ys}$$

$$C_2 = \left\{ D_{i,T_k,id_l}{}^{H_1\left(M||\sigma_{id_l}\right)^s + r} \right\} = \left\{ \left(g^{\dfrac{q_x(0).p_x(0)}{t_i}} \right)^{H_1\left(M||\sigma_{id_l}\right)^s + r} \right\} \tag{1}$$

The signature can be denoted by $v = \{C_1, C_2, C_3, M, \sigma_{id_l}, g^s, H_1(M||\sigma_{id_l})^s\}$. The signer sends v to the receiver.

Verify: After receiving the signature $v = \{C_1, C_2, C_3, M, \sigma_{id}, g^s, H_1(M||\sigma_{id})^s\}$ from the signer, receiver firstly calculates: $\hat{e}(H_1(M||\sigma_{id})^s, g) = \hat{e}(H_1(M||\sigma_{id}), g^s)$.

If the equation is set up, then calculates *VerityNode* $(x, PK, v) = \hat{e}(g^{t_i}, C_2)$.

If x is a leaf node, then the calculation process is as follows:

$$VerityNode(x,PK,v) = \begin{cases} \hat{e}(g^{t_i},C_2) = \hat{e}\left(g^{t_i},\left(g^{\frac{q_x(0).p_x(0)}{t_i}}\right)^{H_1\left(M||\sigma_{id_l}\right)^s+r}\right) \\ = \hat{e}\left(g^{t_i},g^{\frac{q_x(0).p_x(0)}{t_i}}\right)^{H_1\left(M||\sigma_{id_l}\right)^s+r} \\ = \hat{e}\left(g^{p_x(0)},g^{q_x(0)}\right)^{H_1\left(M||\sigma_{id_l}\right)^s}\cdot\hat{e}\left(g^{r\cdot p_x(0)},g^{q_x(0)}\right) \\ \perp, otherwise \end{cases} \tag{2}$$

All the value calculated from $VerityNode(x,PK,v)$ will be stored as F_z. For any $F_z \neq 0$, the algorithm calculates F_{root} (the value of root node) using Lagrange interpolation method.

If x is a non-leaf node, z is the child node of x,then the value of $VerityNode(x,PK,v)$ is calculated as follows:

Let $i = index(z), S'_x = \{index(z) : z \in S_x\}$

$$\begin{aligned} F_x &= \prod_{z\in S_x} F_z{}^{\Delta_{i,Sx'}(0)} \\ &= \prod_{z\in S_x}\hat{e}\left(\left(g^{p_z(0)}\right)^{\Delta_{i,Sx'}(0)},g^{q_z(0)^{\Delta_{i,Sx'}(0)}}\right)^{H_1\left(M||\sigma_{id_l}\right)^s+r} \\ &= \prod_{z\in S_x}\hat{e}\left(\left(g^{p_{parent}(z)^{(index(z))}}\right)^{\Delta_{i,Sx'}(0)},\right. \\ &\qquad \left.\left(g^{q_{parent}(z)^{(index(z))}}{}^{\Delta_{i,Sx'}(0)}\right)\right)H_1(M||\sigma_{id_l})^s+r \\ &= \prod_{z\in S_x}\hat{e}\left(\left(g^{p_x(i)}\right)^{\Delta_{i,Sx'}(0)},\left(g^{q_x(i)\,\Delta_{i,sx'}(0)}\right)\right)^{H_1\left(M||\sigma_{id_l}\right)^s+r} \\ &= \hat{e}\left(g^{p_x(0)},g^{q_x(0)}\right)^{H_1\left(M||\sigma_{id_l}\right)^s}\cdot\hat{e}\left(g^{r\cdot p_x(0)},g^{q_x(0)}\right) \end{aligned} \tag{3}$$

Then, the algorithm calculates F_{root} (the value of root node) by recursive function.

Finally, the algorithm verifies if:

$$F_{root} = C_1{}^{H_1\left(M||\sigma_{id_l}\right)^s}\cdot C_0 \tag{4}$$

If the equation is set up, then the signature is a legal one and the signer's attributes satisfies the claimed structure.

Trace: The tracer searches the list recording the relationship between $id_l – \sigma_{id_l}$, then the identity of the signer can be pinpointed.

Security proof and performance evaluation

Correctness proof

To calculate the value of F_{root}, it is essential to obtain the value of leaf nodes which satisfy the access structure.

Since the value of $VerityNode(x,PK,v) = \hat{e}\left(g^{p_x(0)},g^{q_x(0)}\right)^{H_1\left(M||\sigma_{id_l}\right)^s+r}$, whether x is a leaf node or non-leaf node, consequently, the value of root node F_{root} can be calculated by:

$$\begin{aligned} F_{root} &= \hat{e}\left(g^{p_{root}(0)},g^{q_{root}(0)}\right)^{H_1\left(M||\sigma_{id_l}\right)^s+r} \\ p_{root}(0) &= s, q_{root}(0) = y \end{aligned} \tag{5}$$

Consequently, if the signature is valid then the equation is set up:

$$\begin{aligned} F_{root} &= \hat{e}(g^s,g^y)^{H_1\left(M||\sigma_{id_l}\right)^s+r} \\ &= \left(\hat{e}(g,g)^{ys}\right)^{H_1\left(M||\sigma_{id_l}\right)^s+r} \\ &= C_0\cdot C_1{}^{H_1\left(M||\sigma_{id_l}\right)^s} \end{aligned} \tag{6}$$

Unforgeability

Theorem 1

If our scheme can be broken by an *Adversary* then it can be constructed that a simulator with a non- negligible advantage solves the DBDH problem successfully.

Proof

In the challenge game, if an *Adversary* can break our scheme with advantage ε, then a simulator be constructed to solve the DBDH problem with an advantage of $\frac{\varepsilon}{2}$.

The construction process of the simulator is as follows:

Phase 1 *Setup*:

Challenger sets the parameters as follows:

Defines a global attribute set $U = \{1, 2, \ldots n\}$.

Denotes id_l to be the unique identifier of *Adversary*.

Defines G_1 and G_2 be two cyclic groups of prime order p. The generator of G_1 is denoted by g.

Defines a bilinear paring $\hat{e} : G_1 \times G_1 \rightarrow G_2$.

Picks $\mu \in \{0, 1\}$, $a, b, c, z \in Z_p$.

$$\begin{cases} (A,B,C,Z) = \left(g^a,g^b,g^c,\hat{e}(g,g)^{abc}\right) & if\ \mu = 0 \\ (A,B,C,Z) = \left(g^a,g^b,g^c,\hat{e}(g,g)^z\right) & if\ \mu = 1 \end{cases} \tag{Let}$$

The aim of simulator is to output a value μ^* as a guess of μ.

The simulator runs *Adversary* as sub-program and plays the role of *Challenger*.

Adversary defines attribute set γ.

Simulator randomly chooses $r_i, \beta_i \in Z_p$ and sets PK to be $Y = \hat{e}\left(g^a,g^b\right) = \hat{e}(g,g)^{ab}, T_i = \begin{cases} g^{r_i}, & if\ i\in\gamma \\ g^{b\beta_i}, & if\ i\notin\gamma \end{cases}$

Phase 2 *Queries* :

Private key generation queries:When *Adversary* asks a *Private key generation* query for access structure T_k, simulator responds as follows:

Chooses two polynomial q_x, p_x for each node x in the access structure T_k and sets the degree d_x of q_x, p_x to be one less than the threshold value k_x of that node ($d_x = k_x - 1$). Besides, simulator sets $q_{root}(0) = y, p_{root}(0) = s$. For any other node (except root node) in the access tree, let $q_x(0) = q_{parent(x)}^{index(x)}$. Then simulator calculates the private key and sends D_{i,T_k,id_l} back to *Adversary*.The format of D_{i,T_k,id_l} is as follows:

$$D_{i,T_k,id_l} = \begin{cases} g^{\frac{q_x(0)p_x(0)}{t_i}} = g^{\frac{q_x(0)p_x(0)}{r_i}}, if\ att(x) \in \gamma \\ g^{\frac{q_x(0)p_x(0)}{t_i}} = g^{\frac{q_x(0)p_x(0)}{b\beta_i}}, if\ att(x) \notin \gamma \end{cases}$$

Sign queries: When *Adversary* asks a *Sign* query for access structure T_k and a plaintext M, simulator responds as follows:

Firstly, simulator calculates the private key $D_{i,T_k,\sigma_{id_l}}$.

Obtaining the private key of access structure T_k by running

Randomly chooses $r \in Z_p^*$, let $\sigma_{id_l} = (g^s)^{id_l}$, then calculates:

$$C_0 = \hat{e}(g,g)^{abrs}, C_1 = \hat{e}(g,g)^{abs},\ C_2 = \left\{ D_{i,T_k,id_l}{}^{H_1\left(M||\sigma_{id_l}\right)^s + r} \right\} \tag{7}$$

Then signer outputs $v = \left\{ C_1, C_2, C_3, M, \sigma_{\sigma_{id_l}}, g^s, H_1\left(M||\sigma_{\sigma_{id_l}}\right)^s \right\}$ and sends it to *Adversary*.

It can be seen that the simulator is consistent with our scheme.

Phase 3:*Challenge*:

Adversary outputs a new signature v_l with a challenging access structure T_l and plaintext M.

Adversary cannot make *Sign* queries about T_l and v_l is not gained by a previous *Sign* query.

Adversary forges the signature as the following process:

Chooses a polynomial p_{lx}.Sets $p_{lroot}(0) = s_l$. Picks $r_l \in Z_p$, returns g^{s_l} to simulator.

Simulator chooses sends $\sigma_{\sigma_{id_l}} = (g^{s_l})^{id}$ back to *Adversary*.

Adversary forges the private key $D_{i,T_l,\sigma_{id_l}}{}^*$ of the access structure T_l.

$$D_{i,T_l,\sigma_{id_l}}{}^* = \begin{cases} g^{\frac{q_x{}^*(0)p_{lx}(0)}{t_i}} = g^{\frac{q_x{}^*(0)p_{lx}(0)}{r_i}}, if\ att(x) \in \gamma \\ g^{\frac{q_x{}^*(0)p_{lx}(0)}{t_i}} = g^{\frac{q_x{}^*(0)p_{lx}(0)}{b\beta_i}}, if\ att(x) \notin \gamma \end{cases} \tag{8}$$

Adversary calculates $v_l = \left\{ C_0, C_1, C_2, M, g^{s_l}, \sigma_{\sigma_{id_l}}, H_1\left(M||\sigma_{\sigma_{id_l}}\right)^{s_l} \right\}$:

$$\begin{aligned} & C_0 = \hat{e}(g,g)^{abs_l}, C_1 = \hat{e}(g,g)^{ab.r_l s_l} \\ & C_2 = D_{i,T_l,\sigma_{id_l}}{}^{r_l + H_1\left(M||\sigma_{id_l}\right)^{s_l}} \\ & \quad = \begin{cases} \left(g^{\frac{q_x{}^*(0)p_{lx}(0)}{r_i}} \right)^{r_l + H_1\left(M||\sigma_{id_l}\right)^{s_l}} & , if\ att(x) \in \gamma \\ \left(g^{\frac{q_x{}^*(0)p_{lx}(0)}{b\beta_i}} \right)^{r_l + H_1\left(M||\sigma_{id_l}\right)^{s_l}} & , if\ att(x) \notin \gamma \end{cases} \end{aligned} \tag{9}$$

Verify :

Simulator firstly calculates $VerityNode(x, PK, v_l) = \hat{e}(g^{r_i}, C_2)$.

$$\begin{aligned} & VerityNode(x, PK, v_l) \\ & = \begin{cases} \hat{e}\left(g^{r_i}, \left(g^{\frac{q_x{}^*(0)p_{lx}(0)}{r_i}} \right)^{r_l + H_1\left(M||\sigma_{id_l}\right)^{s_l}} \right), \\ \quad if\ att(x) \in \gamma \\ \hat{e}\left(g^{b\beta_i}, \left(g^{\frac{q_x{}^*(0)p_{lx}(0)}{b\beta_i}} \right)^{r_l + H_1\left(M||\sigma_{id_l}\right)^{s_l}} \right), \\ \quad if\ att(x) \notin \gamma \end{cases} \\ & = \hat{e}\left(g^{p_{lx}(0)}, g^{q_x{}^*(0)} \right)^{r_l + H_1\left(M||\sigma_{id_l}\right)^{s_l}} \end{aligned} \tag{10}$$

By recalling the recursive function, the value of root node F_{root} can be calculated by:

$$\begin{aligned} F_{root} & = \hat{e}\left(g^{p_{lroot}(0)}, g^{q_{root}{}^*(0)} \right)^{r_l + H_1\left(M||\sigma_{id_l}\right)^{s_l}} \\ & = \hat{e}\left(g^{s_l}, g^{q_{root}{}^*(0)} \right)^{r_l + H_1\left(M||\sigma_{id_l}\right)^{s_l}} \\ & = \left(\hat{e}(g,g)^{s_l q_{root}{}^*(0)} \right)^{r_l + H_1\left(M||\sigma_{id_l}\right)^{s_l}} \end{aligned} \tag{11}$$

According to the setting in *Setup* phase, let $s_l = c$, $r_l + H_1 (M||\sigma_{id_l})^{s_l} = f$, then

$$\begin{aligned} F_{root} & = \left(\hat{e}(g,g)^{s_l \cdot q_{root}{}^*(0)} \right)^{r_l + H_1\left(M||\sigma_{id_l}\right)^{s_l}} \\ & = \begin{cases} \left(\hat{e}(g,g)^{abc} \right)^f, if\ \mu = 0 \\ \left(\hat{e}(g,g)^z \right)^f, if\ \mu = 1 \end{cases} \end{aligned} \tag{12}$$

Then we will discuss the advantage of simulator in breaking the DBDH assumption.

When $\mu = 0$, *Adversary* forges the signature successfully. According to our assumption, the probability of this incident is $\frac{1}{2}+\varepsilon$. When *Adversary* successfully forges the signature, simulator guesses $u^* = 0$ and simulator has a $\frac{1}{2}+\varepsilon$ probability of making the correct guess of u^*. The probability of simulator making the correct judgment can be denoted by:

$$Pr(u^* = u|\mu = 0) = \frac{1}{2}+\varepsilon \tag{13}$$

When $\mu = 1$, *Adversary* fails to forge the signature. The result of output is rejected symbol $\perp$. Under this condition simulator guesses μ^* randomly. The probability of simulator making the correct judgment can be denoted by:

$$Pr(u^* = u|\mu = 1) = \frac{1}{2} \tag{14}$$

As is mentioned above, the advantage of simulator is:

$$\begin{aligned}&\frac{1}{2}Pr(u^* = u|\mu = 0) + \frac{1}{2}Pr(u^* = u|\mu = 1)-\frac{1}{2}\\&=\frac{1}{2}\left(\frac{1}{2}+\varepsilon\right)+\frac{1}{2}\times\frac{1}{2}-\frac{1}{2}\\&=\frac{\varepsilon}{2}\end{aligned} \tag{15}$$

Unforgeability with untrusted authorities and traceability

In attribute based signature mechanism, users' attribute private keys are generated and distributed by AA. Under this circumstance, AA is capable of forging any signatures without being detected. To keep the whole system running safely, AA must be honest and highly protected from being invaded. However, in the complex network systems, there are variety risks which may result in the fault of AA. Consequently, for the sake of safety, AA's rights in the system should be reduced to some extent. In our KP-ABS-UT, the private key used for signature is generated by joint efforts of both AA and user, so it is computational infeasible for AA to forge any signatures which should be signed by legal users. Besides, since different users randomly choose different secret numbers to embed into their initial private keys distributed from AA, it is computational infeasible for a user to personate any other users to forge a legal signature in the system.

Anonymity is another important feature of common attribute based signature mechanism. However, some malicious users may take advantage of this feature to send out illegal information without taking responsibility. To prevent such incidents happening, in our KP-ABS-UT scheme, the signer's identity can be revealed with the introducing of "tracer". When data receivers receive illegal information from a malicious signer, they can send the signature v to tracer. Since tracer has stored $\sigma_{id_l} = (g^s)^{id_l}$ with user's identity, the signer's identity can be located by searching the id_l–σ_{id_l} list. Then tracer calculates $\sigma_{id_l}{}^{-id_l} = \left((g^s)^{id_l}\right)^{-id_l} = g^s$. Since s is the secret number used in the *Sign* algorithm and g^s has been sent to tracer, a signer cannot deny the message he signed before.

Efficiency and comparison analysis

In our KP-ABS-UT scheme, assume "n" to be the amount of attributes which a signer owns. The *Sign* algorithm needs (n + 3) times of exponential operation and 1 hash operation. The computation of *Sign* algorithm is low since it does not need any paring operation. The *Verify* algorithm needs (n + 1) times of paring operation and 1 hash operation. Since the computation cost of paring operation is much more than any other operation, in this paper, we mainly compare the number of paring operation with other ABS schemes [9, 14, 18] with respect to efficiency. The efficiency and performance comparison results are listed below in Table 2.

From the result we can see that the computation cost is lower in our KP-ABS-UT since the number of bilinear paring operation is reduced. Thus our plan has a higher efficiency. Besides, our KP-ABS-UT uses access tree as control structure, each signer's private key corresponds a certain structure. It is computational infeasible for different users to collude their private keys with each other to forge a legal signature. With the introducing of user's secret key component and the entity "tracer", the signature cannot be forged by AA and can be traced to identity if a malicious user releases information illegally. The overall comparison shows our KP-ABS-UT is of better performance and is more appropriate for secure data verification in open network systems such as cloud computing, etc.

Conclusion

In this paper, we propose a KP-ABS with untrusted authorities and traceability, which is also of high efficiency. In our scheme, the signer's private key is composed by two components: one part is distributed by attribute authority and the other part is chosen privately by the signer's self. The signature cannot be forged by any other users including attribute authority. What's more, the identity of signer can be traced. Our scheme is proved to be secure and of better performance with respect to efficiency.

Table 2 Efficiency and performance comparison

Scheme	Access method	Traceability	With untrusted AA	Computation cost
[14]	LSSS	Yes	No	(n + 3) paring
[9]	Threshold	No	No	4 paring
[18]	Threshold	No	No	(n + 4) paring
Our scheme	Access tree	Yes	Yes	(n + 1) paring

Our future work should focus on the attribute revocation and key refreshing in our KP-ABS scheme. Once key exposure happens, although the system can trace the traitor, however, user's private keys still need to be refreshed for the safe of privacy. Consequently, research on the attribute based refreshing mechanisms should be taken into our future research direction. Besides, outsourcing ABS [9] with untrusted attribute authorities also merits attention.

Competing interests

The authors declare that they have no competing financial interests.

Authors' contribution

Dr Hanshu Hong: Participated in the design of scheme and drafted the manuscript. Dr Zhixin Sun: Participated in the performance analysis of the proposed KP-ABS-UT. All authors read and approved the final manuscript.

Acknowledgement

This research is supported by the National Natural Science Foundation of China (60973140, 61170276, 61373135).

Received: 17 November 2015 Accepted: 6 March 2016
Published online: 12 March 2016

References

1. Sahai, A, Waters, "Fuzzy identity-based encryption". Proc. Int. Conf. EUROCRYPT 2005, pp. 457-473, Aarhus, Denmark: Springer; 2005. http://link.springer.com/chapter/10.1007%2F11426639_27
2. V. Goyal, O. Pandey, A. Sahai and B. Waters, "Attribute Based Encryption for Fine-Grained Access Control of Encrypted Data", In ACM conference on Computer and Communications Security, pp. 89-98. 2006.
3. Attrapadung N, Libert B, De Panafieu E. Expressive key-policy attribute-based encryption with constant-size ciphertexts, Public Key Cryptography—PKC 2011, vol. 6571 of LNCS. Taormina, Italy: Springer, pp. 90-108, 2011.
4. Bethencourt J, Sahai A, Waters B (2007) Ciphertext-Policy Attribute Based Encryption. Proceedings of the 2007 IEEE Symposium on Security and Privacy, Washington DC, pp 321–334
5. H. K. Maji, M. Prabhakaran, and M. Rosulek, "Attribute-based signatures: Achieving attribute-privacy and collusion-resistance," IACR Cryptology ePrint Archive, pp. 328 – 351, 2008.
6. B. Z. J. S. Dan Cao, Xiaofeng Wang and Q. Hu, "Mediated attribute based signature scheme supporting key revocation," in Proceedings of the 8th International Conference on Information Science and Digital Content Technology. Jeju Island, Korea: IEEE; 2012; 277–282.
7. S.Q. Guo and Y.P. Zeng, "Attribute-based signature scheme", In International Conference on Information Security and Assurance, pp.509-511, 2008.
8. H. K. Maji, M. Prabhakaran, and M. Rosulek, "Attribute-based signatures", in CT-RSA, 2011, pp. 376–392, 2011.
9. Chen X, Li J, Huang X (2014) Secure Outsourced Attribute-Based Signatures". IEEE Transactions on Parallel and Distributed Systems 25(12):3285–3294
10. S. Shahandashti and R. Safavi-Naini, "Threshold attribute-based signatures and their application to anonymous credential systems," Progress in Cryptology–AFRICACRYPT 2009, pp. 198–216, 2009.
11. Kumar S, Agrawal S, Balaraman S, Rangan C (2011) Attribute Based Signatures for Bounded Multi-level Threshold Circuits. Public Key Infrastructures, Services and Applications (EuroPKI 2010), Berlin Heidelberg, pp 141–154
12. J. Herranz, F. Laguillaumie, B. Libert, "Short attribute based signatures for threshold predicates", Topics in Cryptology–CT-RSA 2012, pp. 51–67, 2012.
13. A. Escala, J. Herranz, and P. Morillo, "Revocable attribute-based signatures with adaptive security in the standard model," in AFRICACRYPT, pp. 224–241, 2011.
14. S.L Ding, Y. Zhao, "Efficient Traceable Attribute-Based Signature", in IEEE 13th International Conference on Trust, Security and Privacy in Computing and Communications, pp 582-589, 2014.
15. Li J, Kim K (2010) Hidden attribute-based signatures without anonymity revocation. Inform Sci 180(9):1681–1689
16. Okamoto, T., Takashima, K, "Decentralized attribute-based signatures", Public Key Cryptography-PKC. Nara, Japan: 2013; 125–142.
17. X. Liu, J. Ma, Q. Li, J. Xiong, and F. Huang, "Attribute based multi-signature scheme in the standard model," in Proceedings of the 9th International Conference on Computational Intelligence and Security (CIS'13). Sichuan Province, China: IEEE; 2013; 738–742.
18. Zhiqian Xu, Keith M. Martin, "Anonymous User Revocation for Using Attribute-Based Signature in Cloud Computing". 6th International Conference on Cloud Computing Technology and Science. Singapore: IEEE; 2014; 358-365.
19. Boneh D, Franklin M (2001) Identity-Based encryption from the weil pairing. In: Advances in Cryptology-CRYPTO 2001. Springer-Verlag, LNCS 2139, Berlin, Heidelberg, pp 213–229