Journal of Information Processing Vol. 17 216–231 (Sep. 2009)

Regular Paper

A Dynamic Attribute-Based Group Signature Scheme and Its Application in an Anonymous Survey for the Collection of Attribute Statistics

Keita Emura,[†1] Atsuko Miyaji[†1] and Kazumasa Omote[†1]

Recently, cryptographic schemes based on the user's attributes have been proposed. An Attribute-Based Group Signature (ABGS) scheme is a kind of group signature scheme, where a user with a set of attributes can prove anonymously whether she has these attributes or not. An access tree is applied to express the relationships among some attributes. However, previous schemes did not provide a way to change an access tree. In this paper, we propose a dynamic ABGS scheme that can change an access tree. Our ABGS is efficient in that re-issuing of the attribute certificate previously issued for each user is not necessary. The number of calculations in a pairing does not depend on the number of attributes in both signing and verifying. Finally, we discuss how our ABGS can be applied to an anonymous survey for collection of attribute statistics.

1. Introduction

User identities (such as name, e-mail address and so on) are often used to access several information sources, and moreover as their public keys in Identity-Based Encryption (IBE) schemes[6),10)]. An encryptor can restrict a decryptor to indicate the identity of the decryptor. Recently, cryptographic schemes based on the user's attributes have been proposed. A user has not only his identity but also some attributes such as gender, age, affiliation, and so on. An Attribute-Based Encryption (ABE) is an encryption scheme, where users with some attributes can decrypt any ciphertext associated with these attributes. The first proposed ABE[32)] was inspired by IBE. In ABE schemes, an encryptor can indicate many decryptors by assigning common attributes of these decryptors. There are two kinds of ABE: Key-Policy ABE (KP-ABE) and Ciphertext-Policy ABE (CP-ABE). KP-ABE[16),32)] are schemes where each private key is associated with an access structure. In CP-ABE[9),12),15),26),34)] schemes, each ciphertext is associated with an access structure. As an important property of ABE, collusion resistance of secret keys is required. Users cannot generate a new secret key by combining their secret keys even if users collude with each other. This means that an IBE scheme cannot be regarded as an ABE scheme to treat the user's identity associated with an attribute, since collusion resistance of secret keys is not satisfied. There are some extended ABE schemes like the ABE schemes with multi-authority[11),19)], a distributed ABE scheme[22)], and an attribute-based broadcast encryption scheme[20)]. In addition, there are attribute-based cryptographic schemes with anonymity such as a CP-ABE scheme with recipient anonymity[26)], a secret handshake scheme with fuzzy matching[1)].

Attribute-Based Group Signature (ABGS) schemes[17),18)] have been proposed. ABGS schemes[17),18)] are a kind of Group Signature (GS) schemes[5),14),24)], where a user with a set of attributes can prove anonymously whether she has these attributes or not. The first ABGS[18)] has been constructed using Goyal's ABE[16)] and Boneh's GS[5)]. In addition to this, an ABGS scheme with revocation has been proposed[17)]. To the best of our knowledge, these two schemes are the only proposals for an ABGS. As an important property of ABGS, collusion resistance of attribute certificates is required. Users cannot generate a new attribute certificate by combining their attribute certificates even if users collude with each other. A GS scheme cannot be regarded as an ABGS scheme to treat a signatory group associated with an attribute. For example, a user with the membership certificate of a group A and a user with the membership certificate of a group B can make a group signature of the groups A and B when users collude with each other. Some GSs treat plural groups such as multi-group signature[3)], hierarchical group signature[33)] and sub-group signature[27)]. A multi-group signature[3)] considers that a member belonging to an intersection of two groups can make a group signature corresponding to both groups. A hierarchical group signature[33)] considers hierarchical tree structure and plural group managers. Plural group managers can execute the Join and Revoke algorithms for inferior level signers and the Open algorithm for group signatures made by inferior level signers. In a

†1 School of Information Science, Japan Advanced Institute of Science and Technology

sub-group signature[27], signers belonging to one of the subgroups can make group signatures. The identity of the subgroup used in the signature cannot be determined from the signature. ABGS is a kind of Sub-Group Signature, although ABGS considers plural numbers of *explicit* subgroups, where each subgroup is associated with an attribute.

Usually, users have many kinds of attributes, where some relationships exist among these attributes. An access tree[16)–18] is applied to express these relationships. A trivial ABGS scheme can be constructed to simply combine an ABE scheme and a GS scheme. This construction has been used in previous ABGS schemes[17),18]. However, these schemes did not provide a way to change the relationships among attributes. If an access tree has to be changed (when some threshold values are changed, or some attributes are deleted or added), then a user has only to be re-issued with all attribute certificates to execute the Join algorithm again. In addition to this, the number of calculations in a pairing depends on the number of attributes associated with a signature in these schemes. Our main aim is to solve these problems, namely, the changing of an access tree and the reduction of the computational cost due to bilinear pairing applications in verification.

As an application of ABGS, anonymous survey is known, where an application provider can obtain a collection of user attribute statistical information with relationships among certain attributes without exposing each user's information. An anonymous survey with trusted third parties (TTPs) has been proposed[30]. A user with some attributes sends the distributor a ciphertext encrypted with the public key of the TTP who is in charge of the user's attribute type. However, the distributor cannot verify whether users properly construct the ciphertext or not. An anonymous survey using the open algorithm of Ateniese, et al. GS[2] has been proposed[25]. A distributor can verify whether users properly make the ciphertext or not, to verify the validity of group signatures. Because one attribute certificate is issued for an attribute type, it is difficult for the relationships among some attributes to be handled in the statistical information. Anonymous survey based on ABGS can solve these problems. However, even if anonymous survey is realized by using previous ABGS schemes, attributes and relationships among these attributes can be determined only once, although, for each survey, a different relationship has to be treated. It is desirable that an attribute-based scheme treats the changing of the relationships among attributes without executing any algorithm between users and a group manager.

Our Contribution : In this paper, we propose a dynamic ABGS scheme that can change an access tree when some threshold values are changed, or some attributes are deleted. To achieve the dynamic property, a Bottom-Up Approach construction is introduced, where all secret values are chosen for each attribute associated with each leaf. These secret values of leaves shall not be changed when the access tree is changed. Although there are several protocols based on a tree-based access structure, such as previous ABGS schemes[17),18] and a KP-ABE scheme[16], to the best of our knowledge, our Bottom-Up Approach construction has not been introduced yet. These schemes do not allow the changing of an access tree, namely, they do not guarantee security after the access tree is changed. On the other hand, our scheme guarantees security after the access tree is changed to admit that an adversary can issue the Re-BuildTree oracle to execute the update of an access tree in the security games. Our ABGS is efficient since re-issuing of the attribute certificate that was previously issued for each user is not necessary. When a new attribute *att* is added to an access tree, an attribute certificate corresponding to that specific attribute *att* needs to be issued for the eligible user(s) only. Added to this, the number of calculations in a pairing does not depend on the number of attributes in both signing and verifying. Our scheme is suitable for use in an anonymous survey because the changing of relationships is indispensable in the anonymous survey for the collection of attribute statistics.

Organization : The paper is organized as follows. Definitions are given in Section 2. Our scheme is described in Section 3. Security analysis is performed in Section 4. Efficiency comparisons are presented in Section 5. The application of our ABGS in an anonymous survey for the Collection of Attribute Statistics is demonstrated in Section 6.

2. Definitions

2.1 Bilinear Groups and Complexity Assumptions

Definition 1 (**Bilinear Groups**) We use *bilinear groups* and a *bilinear map* defined as follows:

(1) $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_3$ are cyclic groups of prime order p.
(2) g_1 and g_2 are generators of $\mathbb{G}_1$ and $\mathbb{G}_2$, respectively.
(3) ψ is an efficiently computable isomorphism $\mathbb{G}_2 \to \mathbb{G}_1$ with $\psi(g_2) = g_1$.
(4) e is an efficiently computable bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_3$ with the following properties.
   - *Bilinearity* : for all $u, u' \in \mathbb{G}_1$ and $v, v' \in \mathbb{G}_2$, $e(uu', v) = e(u, v)e(u', v)$ and $e(u, vv') = e(u, v)e(u, v')$.
   - *Non-degeneracy* : $e(g_1, g_2) \neq 1_{\mathbb{G}_3}$ ($1_{\mathbb{G}_3}$ is the $\mathbb{G}_3$'s unit).

Our scheme is based on the *discrete logarithm (DL)*, *q-strong Diffie-Hellman (q-SDH)*[4] and *eXternal Diffie-Hellman (XDH)*[5] assumptions. For the security parameter k, let $\epsilon = \epsilon(k)$ be a negligible function, namely for every polynomial $poly(\cdot)$ and for sufficiently large k, $\epsilon(k) < 1/poly(k)$.

Definition 2 (**DL assumption**) The DL problem in $\mathbb{G}_2$ is defined as follows: given a $(g = (g')^{\xi}, g') \in \mathbb{G}_2^2$ as input, where $\xi \in \mathbb{Z}_p^*$, which outputs a value ξ. An algorithm $\mathcal{A}$ has an advantage ϵ in solving the DL problem in $\mathbb{G}_2$ if $\Pr[\mathcal{A}(g, g') = \xi] \geq \epsilon$. We say that the DL assumption holds in $\mathbb{G}_2$ if no PPT algorithm has an advantage of at least ϵ in solving the DL problem in $\mathbb{G}_2$.

Definition 3 (**q-SDH assumption**) The q-SDH problem in $(\mathbb{G}_1, \mathbb{G}_2)$ is defined as follows: given a $(q + 2)$ tuple $(g, g', (g')^{\xi}, \cdots, (g')^{\xi^q})$ as input, where $g' \in \mathbb{G}_2$, $g = \psi(g') \in \mathbb{G}_1$, $\xi \in \mathbb{Z}_p^*$, which outputs a tuple $(x, g^{1/(\xi+x)})$, where $x \in \mathbb{Z}_p^*$. An algorithm $\mathcal{A}$ has an advantage ϵ in solving the q-SDH problem in $(\mathbb{G}_1, \mathbb{G}_2)$ if $\Pr[\mathcal{A}(g, g', (g')^{\xi}, \cdots, (g')^{\xi^q}) = (x, g^{1/(\xi+x)})] \geq \epsilon$. We say that the q-SDH assumption holds in $(\mathbb{G}_1, \mathbb{G}_2)$ if no PPT algorithm has an advantage of at least ϵ in solving the q-SDH problem in $(\mathbb{G}_1, \mathbb{G}_2)$.

Definition 4 (**DDH assumption**) The DDH problem in $\mathbb{G}_1$ is as follows: given a tuple $(g, g', g^u, (g')^v)$ as input, where $g, g' \in \mathbb{G}_1$ and $u, v \in \mathbb{Z}_p^*$, which outputs 1 if $u = v$ or 0 otherwise. An algorithm $\mathcal{A}$ has an advantage ϵ in solving the DDH problem in $\mathbb{G}_1$ if $|\Pr[\mathcal{A}(g, g', g^u, (g')^u) = 0] - \Pr[\mathcal{A}(g, g', g^u, (g')^v) = 0]| \geq \epsilon$. We say that the DDH assumption holds in $\mathbb{G}_1$ if no PPT algorithm has an advantage of at least ϵ in solving the DDH problem in $\mathbb{G}_1$.

Definition 5 (**XDH assumption**) Bilinear groups $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_3$ and a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_3$ and an efficiently computable isomorphism $\psi : \mathbb{G}_2 \to \mathbb{G}_1$ with $\psi(g_2) = g_1$ are given. We say that the XDH assumption holds if the DDH problem is hard in $\mathbb{G}_1$.

In this paper, we use the notation according to which, if S is a set, then $x \in_R S$ denotes the operation of picking an element x of S uniformly at random.

2.2 Access Tree

Let $Att = \{att_1, \ldots, att_m\}$ be a set of attributes. For $\Gamma \subseteq 2^{Att} \setminus \{\emptyset\}$, Γ satisfies the *monotone property*: if $^\forall B, C \subseteq Att$, $B \in \Gamma$ and $B \subseteq C$, then $C \in \Gamma$ holds. Let access structures for Att be a set of Γ which satisfies the monotone property. An access tree[16)–18)] T is used for expressing an access structure by using a tree structure. An *access tree* is a tree, where threshold gates are defined on each interior node of the tree, and the leaves are associated with attributes. These attributes are subsets of Att. Let ℓ_x be the number of children of node x, and k_x ($0 < k_x \leq \ell_x$) be the threshold value on the threshold gate of node x. We call the threshold gate "OR gate" when $k_x = 1$, and "AND gate" when $k_x = \ell_x$. The notation $Leaves \models T$ expresses the fact that a set of attributes $Leaves$ satisfies the access tree T.

2.3 Model and Security Definitions

In this subsection, we define the model of an ABGS. An *ABGS* is a kind of GS, where a user U_i with a set of attributes $\Gamma_i \subseteq Att = \{att_1, \ldots, att_m\}$ can prove anonymously whether she has these attributes or not. U_i has a membership certificate A_i and a set of attribute certificates $\{T_{i,j}\}_{att_j \in \Gamma_i}$. U_i makes a group signature associated with $\zeta \subseteq \Gamma_i$. Usually, for a set of attributes Att, we construct an access tree to consider all relationships among these attributes. However, the access tree is changed when some threshold values are changed, or some attributes are deleted. Therefore, we define the model of the ABGS accepting a change of an access tree.

Let GM be the *group manager*, k the *security parameter*, $params$ the *system parameter*, $Att = \{att_1, \ldots, att_m\}$ the *a set of attributes*, T_r the *r-th access tree with a set of attributes* $\{att\}$, where $att \in Att$ is assigned on each leaf,

$\mathcal{T}_r$ the public values associated with T_r, gpk the *group public key*, ik the *group secret key* which is used for issuing a membership certificate and making $\mathcal{T}_r$, ok the *opening key* which is used for the opening procedure to reveal the signers' identification from the group signature, (upk_i, usk_i) the *verification/signing key* of a signature scheme $DSig$, sk_i the *member secret key* for U_i $(i = 1, 2, \ldots, n)$, $\Gamma_i \subseteq Att$ attributes of U_i, and reg be the *registration table* for open algorithm. Note that sk_i includes both A_i and $\{T_{i,j}\}_{att_j \in \Gamma_i}$. In the Join algorithm, we use the notation $\mathsf{Join}(\langle \text{input of } GM \rangle, \langle \text{input of user} \rangle)$.

Definition 6 ABGS

- $\mathsf{Setup}(1^k)$: This algorithm takes the security parameter k as an input, and returns the system parameter *params*.
- $\mathsf{KeyGen}(params)$: This algorithm takes as input *params*, and returns the group public key gpk, the group secret key ik, the opening key ok and the registration table $reg = \emptyset$.
- $\mathsf{BuildTree}(params, ik, T_r)$: This algorithm takes as input *params*, ik and the r-th access tree T_r whose leaves are associated with a subset of Att, and returns $\mathcal{T}_r$.
- $\mathsf{Join}(\langle params, gpk, ik, upk_i, \Gamma_i \rangle, \langle params, gpk, upk_i, usk_i \rangle)$: This algorithm takes as input *params*, gpk, ik, upk_i and U_i's attributes Γ_i from GM, and *params*, gpk, upk_i and usk_i from U_i, and returns the member secret key sk_i and reg.
- $\mathsf{Sign}(param, gpk, sk_i, M, \zeta_i, \mathcal{T}_r)$: Let $\zeta_i \subseteq \Gamma_i$ be a set of attributes such that $\zeta_i \models T_r$. This algorithm takes as input *params*, gpk, sk_i, a message M, ζ_i and $\mathcal{T}_r$, and returns a group signature σ associated with ζ_i.
- $\mathsf{Verify}(param, gpk, M, \sigma, \zeta, \mathcal{T}_r)$: This algorithm takes as input *params*, gpk, M, σ, ζ and $\mathcal{T}_r$, and returns 1 if and only if σ is a valid signature.
- $\mathsf{Open}(param, gpk, ok, \sigma, \zeta, \mathcal{T}_r, M, reg)$: This algorithm takes as input *params*, gpk, ok, σ, ζ, $\mathcal{T}_r$, M and reg, and returns the signer's identity i. If the signer is not included in reg, then this algorithm returns 0.

If the access tree T_r is changed to T_{r+1}, then GM runs $\mathtt{BuildTree}(params, ik, T_{r+1})$, and opens $\mathcal{T}_{r+1}$, which is the public information associated with T_{r+1}. When a new attribute att_{m+1} is added to an access tree, an attribute certificate corresponding to that specific attribute att_{m+1} needs to be issued for the eligible user(s) only.

Definition 7 `Anonymity` : *Anonymity* requires that for all PPT $\mathcal{A}$, the probability that $\mathcal{A}$ wins the following game is negligible.

- `Setup`: Let T_0 be the initial access tree. The challenger runs $\mathtt{KeyGen}(params)$, and obtains gpk, ik and ok. The challenger runs $\mathsf{BuildTree}(params, ik, T_0)$, and obtains $\mathcal{T}_0$. $\mathcal{A}$ is given *params*, gpk, $\mathcal{T}_0$ and ik.
- Phase1: $\mathcal{A}$ can send these queries as follows:
  - `Join` : $\mathcal{A}$ requests the join procedure for honest member U_i. $\mathcal{A}$ plays the role of corrupted GM on these queries.
  - `Signing` : $\mathcal{A}$ requests a group signature σ for all messages M, and all members U_i with a set of attributes $\zeta_i \subseteq \Gamma_i$.
  - `Corruption` : $\mathcal{A}$ requests the secret key sk_i for all members U_i.
  - `Open` : $\mathcal{A}$ requests the signer's identity with a message M and a valid signature σ.
  - `Re-BuildTree` : $\mathcal{A}$ sends an access tree T_r. The challenger returns public values $\mathcal{T}_r$.
- `Challenge`: $\mathcal{A}$ outputs M^*, non-corrupted users U_{i_0}, U_{i_1} and ζ. Note that $\zeta \subseteq \Gamma_{i_0}$, $\zeta \subseteq \Gamma_{i_1}$ and $\zeta \models T^*$, where T^* is the access tree on the challenge phase. The challenger uniformly selects $b \in_R \{0,1\}$, and responds with a group signature on M^* by group member U_{i_b}.
- Phase2: $\mathcal{A}$ can make the Signing, Corruption, Open, Join and Re-BuildTree queries. Note that Corruption queries include both U_{i_0} and U_{i_1}.
- `Output`: $\mathcal{A}$ outputs a bit b', and wins if $b' = b$.

The advantage of $\mathcal{A}$ is defined as $Adv^{anon}(\mathcal{A}) = |\Pr(b = b') - \frac{1}{2}|$.

In Join queries, $\mathcal{A}$ can play the role of corrupted GM (the same as in SndToU oracle[7)]). In addition, we consider the Anonymity for *Key-Exposure*, namely, corruption queries for U_{i_0} and U_{i_1} can be admitted in Phase 2. Even after a secret key is exposed, signatures produced by the member before key-exposure remain anonymous. A similar definition of our key-exposure has been given[8)] for the ring signature scheme. Our definition is the *CCA-anonymity model*[5),14)], namely, open queries in the Anonymity game can be admitted.

Definition 8 *Traceability* requires that for all PPT $\mathcal{A}$, the probability that

$\mathcal{A}$ wins the following game is negligible.

- `Setup`: Let T_0 be the initial access tree. The challenger runs `KeyGen`$(params)$, and obtains gpk, ik and ok. The challenger runs $\mathsf{BuildTree}(params, ik, T_0)$, and obtains $\mathcal{T}_0$. $\mathcal{A}$ is given $params$, gpk, $\mathcal{T}_0$ and ok.
- `Queries`: $\mathcal{A}$ can issue the Signing, Corruption, Join and Re-BuildTree queries. All queries are the same as in the Anonymity game, except Join.
  - `Join` : $\mathcal{A}$ requests the Join procedure for corrupted member U_i.
- `Output`: $\mathcal{A}$ outputs a message M^*, σ^* and ζ^*. T^* is the access tree in this phase, and $\mathcal{T}^*$ is the public information associated with T^*.

$\mathcal{A}$ wins if (1) `Verify`$(params, gpk, M^*, \sigma^*, \zeta^*, \mathcal{T}^*) = 1$, (2) $\mathsf{Open}(params, gpk, ok, \sigma^*, \zeta^*, \mathcal{T}^*, M^*, reg) = 0$, and (3) $\mathcal{A}$ has not obtained σ^* in Signing queries on M^*, ζ^* and T^*. The advantage of $\mathcal{A}$ is defined as the probability that $\mathcal{A}$ wins.

In Join queries, $\mathcal{A}$ can play the role of corrupted users (the same as in SndToI oracle[7]).

Definition 9 *Collusion resistance* requires that for all PPT $\mathcal{A}$, the probability that $\mathcal{A}$ wins the following game is negligible.

- `Setup`: Let T_0 be the initial access tree. The challenger runs `KeyGen`$(params)$, and obtains gpk, ik and ok. The challenger runs $\mathsf{BuildTree}(params, ik, T_0)$, and obtains $\mathcal{T}_0$. $\mathcal{A}$ is given $params$, gpk and $\mathcal{T}_0$.
- `Queries`: $\mathcal{A}$ can issue the Signing, Corruption, Join and Re-BuildTree queries. All queries are the same as in the Anonymity game, except Join.
  - `Join` : $\mathcal{A}$ requests the Join procedure for corrupted member U_i.
- `Output`: Finally, $\mathcal{A}$ outputs M^*, σ^* and ζ^*. T^* is the access tree in this phase, and $\mathcal{T}^*$ is the public information associated with T^*.

$\mathcal{A}$ wins if (1) `Verify`$(params, gpk, M^*, \sigma^*, \zeta^*, \mathcal{T}^*) = 1$, and (2) $\mathcal{A}$ has not obtained attribute certificates associated with ζ^* corresponding to a single user.

This property indicates that, for example, there are two users U_{i_0} and U_{i_1} with $\{T_{i_0,j}\}_{att_j \in \Gamma_{i_0}}$ and $\{T_{i_1,j}\}_{att_j \in \Gamma_{i_1}}$, respectively. We assume that $\Gamma_{i_0} \subset \zeta^* \wedge \Gamma_{i_0} \neq \zeta^*$, $\Gamma_{i_1} \subset \zeta^* \wedge \Gamma_{i_1} \neq \zeta^*$, and that $\zeta^* \subseteq \Gamma_{i_0} \cup \Gamma_{i_1}$ hold. Then U_{i_0} and U_{i_1} cannot make a valid group signature with ζ^* even if U_{i_0} and U_{i_1} collude with each other.

Definition 10 *Non-Frameability* requires that for all PPT $\mathcal{A}$, the probability that $\mathcal{A}$ wins the following game is negligible.

- `Setup`: Let T_0 be the initial access tree. The challenger runs `KeyGen`$(params)$, and obtains gpk, ik and ok. The challenger runs $\mathsf{BuildTree}(params, ik, T_0)$, and obtains $\mathcal{T}_0$. $\mathcal{A}$ is given $params$, gpk, $\mathcal{T}_0$, ik and ok.
- `Queries`: $\mathcal{A}$ can issue the Signing, Corruption, Join and Re-BuildTree queries. All queries are the same as in the Anonymity game.
- `Output`: Finally, $\mathcal{A}$ outputs a message M^*, an honest member U_{i^*}, σ^* and ζ^*. T^* is the access tree in this phase, and $\mathcal{T}^*$ is the public information associated with T^*.

$\mathcal{A}$ wins if (1) `Verify`$(params, gpk, M^*, \sigma^*, \zeta^*, \mathcal{T}^*) = 1$, (2) σ^* opens to an honest member U_{i^*}, (3) $\mathcal{A}$ has not obtained σ^* in Signing queries on M^*, U_{i^*} and ζ^*, and (4) $\mathcal{A}$ has not obtained sk_{i^*} in Corruption queries on U_{i^*}. The advantage of $\mathcal{A}$ is defined as the probability that $\mathcal{A}$ wins.

3. Proposed Schemes

In this section, an ABGS together with an assignment of secret values to access trees is presented.

3.1 Assignment of Secret Values to Access Trees

In this subsection, we propose the assignment of secret values to access trees. The previous schemes[17),18)] use a *"Top-Down Approach"* construction for access trees (when threshold gates are defined on each interior node of the tree and the leaves are associated with attributes) as follows:

- A secret value of the root node is chosen.
- A polynomial $q_{root}(x)$ of degree "threshold value -1" is defined such that $q_{root}(0)$ equals the secret value of the root node.
- A secret value of a child node is defined such as $q_{root}(index(child))$.
- Secret values of all nodes can be defined to execute this procedure recursively.

If the access tree is changed, then the above Top-Down Approach construction has to be executed again. This means that the secret values that are associated with attributes have to be re-issued to corresponding users, because these values have to be changed. In our proposal, a *"Bottom-Up Approach"* construction is introduced. The order of our construction is different from that of the Top-Down Approach construction, namely, first all secret values are chosen for each attribute associated with each leaf. These secret values of leaves will not be changed when the access tree is changed. Therefore, our ABGS is efficient in that re-issuing of the attribute certificate previously issued for each user is not necessary.

Idea: For a node x associated with the threshold value k_x, $\ell_x - k_x$ *dummy nodes* will be opened, where ℓ_x is the number of children of x. Next, the threshold value is changed from k_x to ℓ_x. Then, all threshold gates become AND gates. Children with k_x or more can compute the secret value of their parent node by using the number of $\ell_x - k_x$ public dummy nodes. We define functions $\mathsf{AddDummyNode}$ which adds dummy nodes to the access tree, $\mathsf{AssignedValue}$ which assigns secret values for nodes on the access tree, and $\mathsf{MakeSimplifiedTree}$ which makes a *simplified tree* associated with a set of leaves. Let *index* be the function which returns the index of the node, and p be a prime number. We assume that T includes Att.

$\langle \mathsf{AddDummyNode}(T) \rangle$: This algorithm takes as input T, and returns the *extended access tree* T^{ext} with dummy nodes on T.

(1) For an interior node x of T, the number of dummy nodes $\ell_x - k_x$ is added to x's children.
(2) The threshold value defined in x is changed from k_x to ℓ_x.
(3) All nodes are assigned unique index numbers.
(4) The resulting tree, called T^{ext}, is output.

Let D_T be a set of dummy nodes determined by $\mathsf{AddDummyNode}$. We assume that T^{ext} includes D_T. Let $s_j \in \mathbb{Z}_p$ be a secret value for an attribute $att_j \in Att$. Let $S = \{s_j\}_{att_j \in Att}$.

$\langle \mathsf{AssignedValue}(p, S, T^{ext}) \rangle$: This algorithm takes as input p, S and T^{ext} and returns a secret value $s_x \in \mathbb{Z}_p$ for each node x of T^{ext}. Let $\{child\}_x$ be the set of node x's children except the dummy nodes, and $\{d\}_x$ be the set of node x's dummy nodes.

(1) For an interior node x of T^{ext}, a polynomial q_x of degree $\ell_x - 1$ is assigned as follows:
  (a) For $att_j \in \{child\}_x$, let q_x be a polynomial of degree at most $\ell_x - 1$ which passes though $(index(att_j), s_j)$, where $s_j \in S$ $(j = 1, 2, \ldots, \ell_x)$.
  (b) For a dummy node $d_j \in \{d\}_x$, the secret value $s_{d_j} := q_x(index(d_j))$ $(j = 1, 2, \ldots, \ell_x - k_x)$ is assigned.
  (c) For x, $s_x := q_x(0)$ is assigned.
(2) Repeat the above procedure up to the root node, $s_T := q_{root}(0)$ is the secret value of T.
(3) Output $\{s_{d_j}\}_{d_j \in D_T}$ and s_T.

$\langle \mathsf{MakeSimplifiedTree}(Leaves, T^{ext}) \rangle$: This algorithm takes as input the set of attributes $Leaves \subseteq Att$ satisfying $Leaves \models T$, and returns the simplified access tree T^{Leaves} (which is the access tree associated with $Leaves$) and a product of Lagrange coefficients Δ_{leaf}.

(1) The set of attributes $\{att_j\}_{att_j \in Att \setminus Leaves}$ are deleted from T^{ext}.
(2) An interior node x has children less than the threshold value (namely, ℓ_x), and is deleted from T^{ext} along with x's descendants.
(3) Let D^{Leaves} be the set of dummy nodes which have remained after (1) and (2), and T^{Leaves} be the access tree after (1) and (2).
(4) For all nodes x of T^{Leaves} except root, we define L_x as follows:
  (a) For x, define the depth 2 subtree of T^{Leaves} with x as leaf node. Let c_x be the set of indices of leaves.
  (b) Compute $L_x := \prod_{k \in c_x \setminus \{index(x)\}} \frac{-k}{index(x) - k}$.
(5) Let $leaf \in \{att_j \in Leaves\} \cup \{d_j \in D^{Leaves}\}$ be a leaf node of T^{Leaves}. For $leaf$, we define Δ_{leaf} as follows:
  (a) Let $Path_{leaf} := \{leaf, parent_1, \ldots, parent_{n_{leaf}} = root\}$ be the set of nodes that appears in the path from $leaf$ to root node.
  (b) Compute $\Delta_{leaf} := \prod_{node \in Path_{leaf} \setminus root} L_{node}$.
(6) Output T^{Leaves}, Δ_j $(att_j \in Leaves)$, Δ_{d_j} $(d_j \in D^{Leaves})$.

Clearly, $\sum_{att_j \in Leaves} \Delta_j s_j + \sum_{d_j \in D^{Leaves}} \Delta_{d_j} s_{d_j} = s_T$ holds.

Example 1 Let $Att = \{A, B, C, D, E, F\}$ and T be a tree defined in **Fig. 1**.

Then, $T^{ext} = \mathsf{AddDummyNode}(T)$ is as follows (See **Fig. 2**). Let each index be assigned by using the depth-first search method. Then $D_T = \{d_1, d_2, d_3, d_4\}$.

Next, we run $\mathsf{AssignedValue}(p, S, T^{ext})$. We introduce the assignment of secret values for the depth 2 subtree of T^{ext} such that A and B are leaves (See **Fig. 3**).

Let $Leaves = \{A, E\}$. Then $Leaves \models T$ holds. The result of $\mathsf{MakeSimplifiedTree}(Leaves, T^{ext})$ is as follows (See **Fig. 4**):

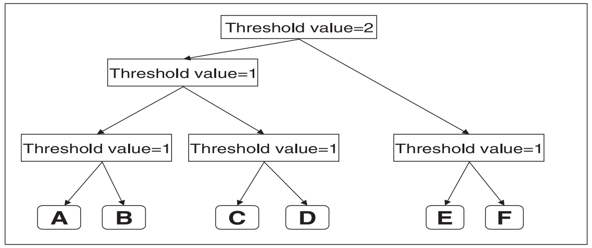

Fig. 1 Access Tree $\boldsymbol{T}$.

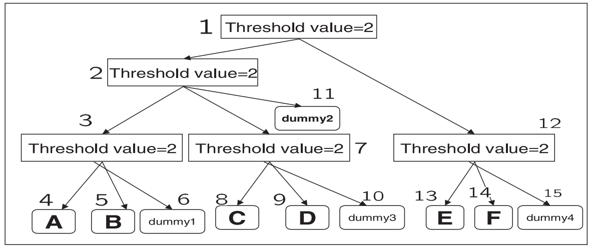

Fig. 2 Extended Access Tree $\boldsymbol{T^{ext}}$.

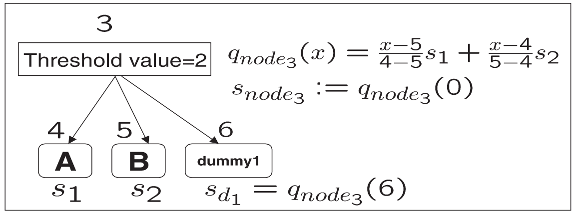

Fig. 3 Assignment of Secret Values on $\boldsymbol{T^{ext}}$.

$D^{\text{Leaves}}=\{d_1, d_2, d_4\}$. $s_{node_3} = L_4 s_1 + L_6 s_{d_1} = \frac{-6}{4-6}s_1 + \frac{-4}{6-4}s_{d_1}$, $s_{node_2} = L_3 s_{node_3} + L_{11}s_{d_2} = \frac{-11}{3-11}s_{node_3} + \frac{-3}{11-3}s_{d_2}$, and $s_{node_2} = L_3(L_4 s_1 + L_6 s_{d_1}) + L_{11}s_{d_2} = (L_4L_3)s_1 + (L_6L_3)s_{d_1} + L_{11}s_{d_2}$ holds. Therefore $s_T = (L_4L_3L_2)s_1 + (L_6L_3L_2)s_{d_1} + (L_{11}L_2)s_{d_2} + (L_{13}L_{12})s_5 + (L_{15}L_{12})s_{d_4} = \Delta_4 s_1 + \Delta_{13}s_5 + \Delta_6 s_{d_1} + \Delta_{11}s_{d_2} + \Delta_{15}s_{d_4}$ holds.

3.2 Proposed Attribute-Based Group Signature Scheme

In this subsection, we propose the ABGS by using our assignment (Section 3.1). Our ABGS uses the Cramer-Shoup encryption scheme[13] for both CCA-anonymity and key-exposure properties, and a concurrently secure Join algorithm[14]. Let $NIZK$ be a *Non-Interactive Zero-Knowledge* proof, SPK a *Signature of Proof of Knowledge*, and *Ext-Commit* be an extractable commitment scheme which uses the Paillier's encryption scheme[28]. *Ext-Commit* is necessary to extract the committed secret value of a corrupted user in the proof of Traceability. Let T_0 be the initial access tree. Note that if an access tree is changed, then GM runs $\texttt{BuildTree}(params, ik, T_{r+1})$, and opens $\mathcal{T}_{r+1}$, which is the public information associated with T_{r+1}.

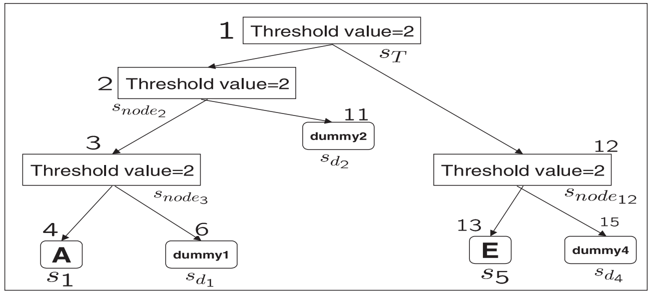

Fig. 4 Simplified Access Tree $\boldsymbol{T^{Leaves}}$.

- $\texttt{Setup}(1^k)$
  - (1) GM selects cyclic groups of $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_3$ with prime order p, an isomorphism $\psi : \mathbb{G}_2 \to \mathbb{G}_1$, a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_3$, and a hash function $\mathcal{H} : \{0,1\}^* \to \mathbb{Z}_p$.
  - (2) GM selects a generator $g_2 \in \mathbb{G}_2$ and $g_3, g_4 \in_R \mathbb{G}_1$, and sets $g_1 = \psi(g_2)$.
  - (3) GM defines $Att = \{att_1, att_2, \ldots, att_m\}$.
  - (4) GM outputs $params = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_3, e, \psi, \mathcal{H}, g_1, g_2, g_3, g_4, Att)$.
- $\texttt{KeyGen}(params)$
  - (1) GM selects $\gamma \in_R \mathbb{Z}_p$, and computes $\omega = g_2^{\gamma}$.
  - (2) GM selects $x_1', x_2', y_1', y_2', z \in_R \mathbb{Z}_p$, and computes $C = g_3^{x_1'} g_4^{x_2'}, D = g_3^{y_1'} g_4^{y_2'}$ and $E = g_3^z$.
  - (3) For $att_j \in Att$, GM selects $s_j \in_R \mathbb{Z}_p^*$, sets $S = \{s_j\}_{att_j \in Att}$, and computes $g_{att_j} = g_2^{s_j}$ $(att_j \in Att)$.

(4) For $att_j \in Att$, GM selects $h_j \in_R \mathbb{G}_2$, and sets $\hat{h}_j = \psi(h_j)$.
(5) GM outputs $ok = (z)$, $gpk = (\omega, C, D, E, \{h_j\}_{j=1}^m, \{g_{att_j}\}_{att_j \in Att})$ and $ik = (\gamma, \{s_j\}_{att_j \in Att})$.

- `BuildTree`$(params, ik, T_0)$
  (1) GM runs $T_0^{ext} = \mathsf{AddDummyNode}(T_0)$ and $\mathsf{AssignedValue}(p, S, T_0^{ext})$, and gets $\{s_{d_j}\}_{d_j \in D_{T_0}}$ and s_{T_0}.
  (2) GM computes $g_{d_j} = g_2^{s_{d_j}}$ $(d_j \in D_{T_0})$ and $v_0 = g_2^{s_{T_0}}$.
  (3) GM outputs $\mathcal{T}_0 = (\{g_{d_j}\}_{d_j \in D_{T_0}}, v_0, T_0^{ext})$.
- `Join`$(\langle params, gpk, ik, upk_i, \Gamma_i\rangle, \langle params, gpk, upk_i, usk_i\rangle)$
  U_i gets $sk_i = ((A_i, x_i, y_i), \{T_{i,j}\}_{att_j \in \Gamma_i})$, where (A_i, x_i, y_i) is a member certificate and $\{T_{i,j}\}_{att_j \in \Gamma_i}$ is the set of attribute certificates.
  (1) U_i picks $y_i \in_R \mathbb{Z}_p$ and computes $c_i = Ext\text{-}Commit(y_i)$, $F_i = E^{y_i}$ and $\pi_1 = NIZK\{y_i : F_i = E^{y_i} \wedge c_i = Ext\text{-}Commit(y_i)\}$.
  (2) U_i sends F_i, c_i and π_1 to GM.
  (3) GM checks π_1. If π_1 is not valid, then abort.
  (4) GM selects $x_i \in_R \mathbb{Z}_p$ and computes $A_i = (g_1 F_i)^{1/(\gamma + x_i)}$, $B_i = e(g_1 F_i, g_2)/e(A_i, w)$, $D_i = e(A_i, g_2)$, $T_{i,j} = A_i^{s_j}$ $(att_j \in \Gamma_i)$, and $\pi_2 = NIZK\{x_i, s_j\ (att_j \in \Gamma_i) : B_i = D_i^{x_i} \wedge T_{i,j} = A_i^{s_j}\ (att_j \in \Gamma_i) \wedge g_{att_j} = g_2^{s_j}\ (att_j \in \Gamma_i)\}$.
  (5) GM sends $A_i, B_i, D_i, \{T_{i,j}\}_{att_j \in \Gamma_i}$ and π_2 to U_i.
  (6) U_i checks π_2. If π_2 is not valid, then abort.
  (7) U_i makes $S_{i,A_i} = DSig_{usk_i}(A_i)$ and sends S_{i,A_i} to GM.
  (8) GM verifies S_{i,A_i} with respect to upk_i and A_i. If S_{i,A_i} is valid, then GM sends x_i to U_i and adds (U_i, A_i) to reg.
  (9) U_i checks the relation $A_i^{(x_i+\gamma)} = g_1 E^{y_i}$ to verify whether $e(A_i, g_2)^{x_i} e(A_i, w) e(E, g_2)^{-y_i} \stackrel{?}{=} e(g_1, g_2)$.

  GM chooses $s_{m+1} \in \mathbb{Z}_p^*$, and computes $g_{att_{m+1}} = g_2^{s_{m+1}}$ when an attribute att_{m+1} is added. Let U_i be issued $T_{i,m+1}$. Then GM computes $T_{i,m+1} = A_i^{s_{m+1}}$ and $\pi_3 = NIZK\{s_{m+1} : T_{i,m+1} = A_i^{s_{m+1}} \wedge g_{att_{m+1}} = g_2^{s_{m+1}}\}$, and sends $T_{i,m+1}$ and π_3 to U_i, and opens $g_{att_{m+1}}$.
- `Sign`$(param, gpk, sk_i, M, \zeta_i, \mathcal{T}_r)$
  A signer U_i signs a message $M \in \{0,1\}^*$ as follows:
  (1) U_i chooses $\zeta_i \subseteq \Gamma_i$ $(\zeta_i \models T_r)$ to associate ζ_i with a group signature. Let $|\zeta_i| = \phi$.
  (2) U_i runs $\mathsf{MakeSimplifiedTree}(\zeta_i, T_r^{ext})$, and gets $T_r^{\zeta_i}$, Δ_j $(att_j \in \zeta_i)$ and Δ_{d_j} $(d_j \in D_r^{\zeta_i})$.
  (3) U_i computes $g_d = \prod_{d_j \in D_r^{\zeta_i}} g_{d_j}^{\Delta_{d_j}}$.
  (4) U_i selects $\alpha, \delta \in_R \mathbb{Z}_p$, and computes $C_1 = A_i E^{\alpha}$, $C_2 = g_3^{\alpha}$, $C_3 = g_4^{\alpha}$ and $C_4 = (CD^{\beta})^{\alpha}$, where $\beta = \mathcal{H}(C_1, C_2, C_3)$.
  (5) U_i computes $CT_j = T_{i,j} \hat{h}_j^{\delta}$ $(att_j \in \zeta_i)$.
  (6) U_i sets $\tau = \alpha x_i + y_i$, and computes $V = SPK\{(\alpha, x_i, \tau, \delta) : \frac{e(C_1,\omega)}{e(g_1,g_2)} = \frac{e(E,g_2)^{\tau} \cdot e(E,\omega)^{\alpha}}{e(C_1,g_2)^{x_i}} \wedge C_2 = g_3^{\alpha} \wedge C_3 = g_4^{\alpha} \wedge C_4 = (CD^{\beta})^{\alpha} \wedge \frac{e(\prod_{att_j \in \zeta_i} CT_j^{\Delta_j}, g_2)}{e(C_1, v_r/g_d)} = \frac{e(\prod_{att_j \in \zeta_i} \hat{h}_j^{\Delta_j}, g_2)^{\delta}}{e(E, v_r/g_d)^{\alpha}}\}(M)$. Concretely, U_i computes V as follows:
    (a) U_i selects $r_{\alpha}, r_{x_i}, r_{\tau}, r_{\delta} \in_R \mathbb{Z}_p$.
    (b) U_i computes $R_1 = \frac{e(E,g_2)^{r_{\tau}} e(E,\omega)^{r_{\alpha}}}{e(C_1,g_2)^{r_{x_i}}}$, $R_2 = g_3^{r_{\alpha}}$, $R_3 = g_4^{r_{\alpha}}$, $R_4 = (CD^{\beta})^{r_{\alpha}}$ and $R_{Att} = \frac{e(\prod_{att_j \in \zeta_i} \hat{h}_j^{\Delta_j}, g_2)^{r_{\delta}}}{e(E, v_r/g_d)^{r_{\alpha}}}$.
    (c) U_i computes $c = \mathcal{H}(gpk, M, \{C_i\}_{i=1}^4, \{CT_i\}_{i=1}^{\phi}, \{R_i\}_{i=1}^4, R_{Att})$.
    (d) U_i computes $s_{\alpha} = r_{\alpha} + c\alpha$, $s_{x_i} = r_{x_i} + cx_i$, $s_{\tau} = r_{\tau} + c\tau$ and $s_{\delta} = r_{\delta} + c\delta$.
  (7) U_i outputs $\sigma = (\{C_i\}_{i=1}^4, c, s_{\alpha}, s_{x_i}, s_{\tau}, s_{\delta}, \{CT_i\}_{i=1}^{\phi})$

  A signer U_i proves the knowledge of $(\alpha, x_i, \tau, \delta)$ which satisfies the 5 above relations described in SPK. The first relation captures whether a signer has a valid membership certificate issued by the Join algorithm or not. The last relation captures whether a signer has valid attribute certificates associated with the set of attributes $\zeta_i \models T_r$ or not.
- `Verify`$(param, gpk, M, \sigma, \zeta, \mathcal{T}_r)$
  A verifier verifies a group signature σ associated with the set of attributes ζ.
  (1) The verifier runs $\mathsf{MakeSimplifiedTree}(\zeta, T_r^{ext})$, and gets T_r^{ζ}, Δ_j $(att_j \in \zeta)$ and Δ_{d_j} $(d_j \in D_r^{\zeta})$. Let $|\zeta| = \phi$.

(2) The verifier computes $g_d = \prod_{d_j \in D_r^{\zeta}} g_{d_j}^{\Delta_{d_j}}$ and $\beta = \mathcal{H}(C_1, C_2, C_3)$.

(3) The verifier computes $\tilde{R}_1 = \frac{e(E,g_2)^{s_\tau} \cdot e(E,\omega)^{s_\alpha}}{e(C_1,g_2)^{s_{x_i}}} \left(\frac{e(g_1,g_2)}{e(C_1,\omega)}\right)^c$, $\tilde{R}_2 = g_3^{s_\alpha}\left(\frac{1}{C_2}\right)^c$, $\tilde{R}_3 = g_4^{s_\alpha}\left(\frac{1}{C_3}\right)^c$, $\tilde{R}_4 = (CD^\beta)^{s_\alpha}\left(\frac{1}{C_4}\right)^c$ and $\tilde{R}_{Att} = \frac{e(\prod_{att_j \in \zeta_i} \hat{h}_j^{\Delta_j}, g_2)^{s_\delta}}{e(E, v_r/g_d)^{s_\alpha}} \left(\frac{e(C_1, v_r/g_d)}{e(\prod_{att_j \in \zeta_i} CT_j^{\Delta_j}, g_2)}\right)^c$.

(4) The verifier checks $c \stackrel{?}{=} \mathcal{H}(gpk, M, gpk, M, \{C_i\}_{i=1}^4, \{CT_i\}_{i=1}^{\phi}, \{\tilde{R}_i\}_{i=1}^4, \tilde{R}_{Att})$.

- `Open`$(param, gpk, ok, \sigma, \zeta, \mathcal{T}_r, M, reg)$

(1) GM verifies the validity of σ by using Verify$(param, gpk, M, \sigma, \zeta, \mathcal{T}_r)$. If σ is not a valid signature, then GM outputs $\perp$.

(2) GM computes $A_i = C_1/C_2^z$.

(3) GM searches A_i from reg, and outputs identity i. If there is no entry in reg, then GM outputs 0.

Our ABGS can be regarded as a GS without having the related part of $s_\delta, CT_1, \ldots, CT_\phi$. Then $\sigma = (C_1, C_2, C_3, C_4, c, s_\alpha, s_{x_i}, s_\tau)$ is a group signature, where $c = \mathcal{H}(gpk, M, C_1, C_2, C_3, C_4, R_1, R_2, R_3, R_4)$. Our GS provides CCA-anonymity, key-exposure and non-frameability. Boneh's GS[5)] (which applied by Khader to propose ABGS) does not provide above properties. This is the reason why we apply the Cramer-Shoup encryption scheme to propose our ABGS.

3.3 Reduce the Authority of the group manager

In this subsection, we describe the authority of GM. In the Join algorithm, GM can obtain all attributes of all group members, since GM knows the attributes of the members when issuing attribute certificates. Therefore, the authority of GM is stronger compared with GM of usual GS schemes[5),14),24)]. There is a ways to distribute the authority of GM. Several separate GMs for each role are defined, namely, an issuer who issues membership certificates and an opener who opens a group signature are defined. Here, we give the detailed way to distribute the authority. An issuer who issues membership certificates, an opener who opens a group signature, and several *Attribute Managers* (AMs) are defined. An AM_k ($k \in \mathbb{N}$) manages a set of attribute $Att_k \subseteq Att$, and issues an attribute certificate associated with $att \in Att_k$. For $k \neq k'$, $Att_k \cap Att_{k'} = \emptyset$ is assumed. Although AM_k obtains attributes Att_k of all group members, AM_k does not obtain attributes $Att_{k'}$ of all group members, where $k' \neq k$. As a classification of dividing AMs, AM_k is defined for a unity of attributes which belong to the same category. For example, we consider unities of attributes "gender" and "age", and a tree-structure is expressed as $(male \vee female) \wedge (10s, \ldots, 80s)$. Then AM_1 manages $\{male, female\}$, and AM_2 manages $\{10s, \ldots, 80s\}$. In KeyGen phase, AM_k chooses $s_j \in \mathbb{Z}_p^*$, where $att_j \in Att_k$. In Join phase, AM_k issues attribute certificates $T_{i,j} = A_i^{s_j}$ for a user U_i, where $att_j \in Att_k \cap \Gamma_i$. This procedure is as follows: Let U_i be issued $T_{i,j}$, where $att_j \in Att_k$. Then AM_k computes $T_{i,j} = A_i^{s_j}$ and $\pi_4 = NIZK\{s_j : T_{i,j} = A_i^{s_j} \wedge g_{att_j} = g_2^{s_j}\}$.

4. Security

In this section, we show that our scheme satisfies anonymity, traceability, collusion resistance, and non-frameability. Let p, q_H and q_S be the order of bilinear groups, and the number of hash queries and signature queries, respectively.

Theorem 1 The proposed scheme satisfies anonymity under the XDH assumption (namely DDH assumption over $\mathbb{G}_1$) in the random oracle model, i.e., $Adv^{anon}(\mathcal{A}) \leq \frac{q_S q_H}{p} + m \cdot \epsilon_{\text{ddh}}$ holds, where ϵ_{ddh} is the DDH-advantage of some algorithms and $m = |Att|$.

Theorem 2 We suppose an adversary $\mathcal{A}$ breaks the Traceability of the proposed scheme with the advantage ϵ. Then, in the random oracle model, we can construct an algorithm $\mathcal{B}$ that breaks the q-SDH assumption with the advantage $\frac{1}{6}(1 - \frac{1}{p})(1 - \frac{q_S q_H}{p})\epsilon$.

Theorem 3 We suppose an adversary $\mathcal{A}$ breaks the non-frameability of the proposed scheme with the advantage ϵ. Then, we can construct an algorithm $\mathcal{B}$ that breaks the DL assumption with the advantage $\frac{1}{12}(1 + \frac{1}{n})(1 - \frac{q_S q_H}{p})\epsilon$.

Theorem 4 The probability that a signature by forged attribute certificates passes the verification, $\Pr(\texttt{Verify}(params, gpk, M, \sigma, \zeta, \mathcal{T}) = 1 \wedge \zeta \not\models T)$, is at most $1/p$.

Theorem 5 Even if some malicious participants $U_{i_1}, \ldots, U_{i_k}$ ($k > 1$) with the set of attributes $\zeta_{i_1}, \ldots, \zeta_{i_k}$ collude, they cannot make a valid signature associated with an attribute tree T_r, where $(\cup_{j=1}^{k} \zeta_{i_j}) \models T_r$ and $\zeta_{i_j} \not\models T_r$ $(j = 1, \ldots, k)$ with non-negligible probability.

We give the proof of Theorem 1 as follows:

Proof 1 We give a proof of anonymity of the proposed scheme under the XDH assumption (namely DDH assumption over $\mathbb{G}_1$), using a sequence of games[31]. Let $\mathcal{C}_j$ and S_j be the challenger on j-th Game and the event that an adversary wins on j-th Game, respectively, and $\sigma^* = (C_1^*, C_2^*, C_3^*, C_4^*, \{CT_j^*\}_{att_j \in \zeta}, V^*)$ be the challenge group signature. Moreover, let q_H and q_S be the number of hash queries and signature queries, respectively, and ϵ_{ddh} be the DDH-advantage of some algorithms. Without loss of generality, $\zeta = \{att_1, att_2, \ldots, att_\phi\}$.

Game 0. This is the original anonymity game defined in Definition 7. Let $params = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_3, e, \psi, \mathcal{H}, g_1, g_2, g_3, g_4, Att)$. $\mathcal{C}_0$ chooses $x_1', x_2', y_1', y_2', z_1, z_2 \in_R \mathbb{Z}_p^*$. Moreover, $\mathcal{C}_0$ computes $C = g_3^{x_1'} g_4^{x_2'}$, $D = g_3^{y_1'} g_4^{y_2'}$ and $E = g_3^{z_1} g_4^{z_2}$. Although R. Cramer and V. Shoup insist that "*the simulator's key generation algorithm is slightly different from the key generation algorithm of the actual cryptosystem*", in the original paper[13], W. Mao shows that "*the E component of the public key construction is perfectly valid*"[21]. Other parameters are chosen the same as for the real scheme setting. $\mathcal{A}$ is given $params$, gpk and ik. $\mathcal{C}_0$ can answer all queries, because $\mathcal{C}_0$ can choose all secret keys (ik, ok, sk_i). Especially, for Re-BuildTree query T_r, $\mathcal{C}_0$ can run $\mathsf{AssignedValue}(p, S, \mathsf{AddDummyNode}(T_r))$, and returns $\mathcal{T}_r = (\{g_{d_j}\}_{d_j \in D_{T_r}}, v_r = g_2^{s_{T_r}}, T_r^{ext})$. $\mathcal{A}$ outputs M^*, U_{i_0}, U_{i_1}, and ζ in the challenge phase. $\mathcal{C}_0$ computes T_ζ from both ζ and T^*, where T^* is the access tree on this phase. Moreover $\mathcal{C}_0$ selects $b \in_R \{0,1\}$, and computes A_{i_b} and $\{T_{i_b,j}\}_{att_j \in \zeta} = \{A_{i_b}^{s_j}\}_{att_j \in \zeta}$ by using ik. $\mathcal{C}_0$ selects $u \in_R \mathbb{Z}_p$ and computes $C_1^* = A_{i_b} E^u$, $C_2^* = g_3^u$, $C_3^* = g_4^u$ and $C_4^* = (CD^{\beta^*})^u$, where $\beta^* = \mathcal{H}(C_1^*, C_2^*, C_3^*)$. $\mathcal{C}_0$ selects $\delta \in_R \mathbb{Z}_p$, and computes $\{CT_j^*\}_{att_j \in \zeta} = \{T_{i_b,j} \hat{h}_j^{\delta}\}_{att_j \in \zeta}$ and $V = SPK\{(u, x_{i_b}, \tau, \delta) : \frac{e(C_1^*, \omega)}{e(g_1, g_2)} = \frac{e(E, g_2)^{\tau} \cdot e(E, \omega)^u}{e(C_1^*, g_2)^{x_{i_b}}} \wedge C_2^* = g_3^u \wedge C_3^* = g_4^u \wedge C_4^* = (CD^{\beta^*})^u \wedge \frac{e(\prod_{att_j \in \zeta} CT_j^{*\,\Delta_j}, g_2)}{e(C_1^*, v^*/g_d)} = \frac{e(\prod_{att_j \in \zeta} \hat{h}_j^{\Delta_j}, g_2)^{\delta}}{e(E, v^*/g_d)^u}\}(M^*)$, where $\tau = u x_{i_b} + y_{i_b}$ and $v^* = g_2^{s_{T^*}}$. Then the adversary's advantage $Adv^{anon}(\mathcal{A})$ is $|\Pr[S_0] - \frac{1}{2}|$.

Game 1. This is the same as Game 0 except SPK V^*, which includes the backpatch of the hash function $\mathcal{H}$, is simulated as follows:

(1) $\mathcal{C}_1$ selects $c^*, s_{x_{i_b}}^*, s_u^*, s_\tau^*, s_\delta^* \in_R \mathbb{Z}_p$.

(2) $\mathcal{C}_1$ computes $R_1^* = \frac{e(E, g_2)^{s_\tau^*} \cdot e(E, \omega)^{s_u^*}}{e(C_1^*, g_2)^{s_{x_{i_b}}^*}} \left(\frac{e(g_1, g_2)}{e(C_1^*, \omega)}\right)^{c^*}$, $R_2^* = g_3^{s_u^*} \left(\frac{1}{C_2^*}\right)^{c^*}$, $R_3^* = g_4^{s_u^*} \left(\frac{1}{C_3^*}\right)^{c^*}$, $R_4^* = (CD^{\beta^*})^{s_u^*} \left(\frac{1}{C_4^*}\right)^{c^*}$ and

$$R_{Att}^* = \frac{e(\prod_{att_j \in \zeta} \hat{h}_j^{\Delta_j}, g_2)^{s_\delta^*}}{e(E, v^*/g_d)^{s_u^*}} \left(\frac{e(C_1^*, v^*/g_d)}{e(\prod_{att_j \in \zeta} CT_j^{*\,\Delta_j}, g_2)}\right)^{c^*}.$$

(3) $\mathcal{C}_1$ defines $c^* := \mathcal{H}(gpk, M, gpk, M, C_1^*, \ldots, C_4^*, CT_1^*, \ldots, CT_\phi^*, R_1^*, \ldots, R_4^*, R_{Att}^*)$.

(4) $\mathcal{C}_1$ outputs $V^* = (c^*, s_{x_{i_b}}^*, s_u^*, s_\delta^*, s_\tau^*)$.

If simulation of the zero knowledge proof fails in signing queries, then $\mathcal{C}_1$ aborts. This probability $\Pr[\mathsf{abort}]$ is at most $q_S q_H / p$[24]. Then $|\Pr[S_0] - \Pr[S_1]| = \Pr[\mathsf{abort}]$ holds.

Game 2. This is the same as Game 1 except C_1^*, C_2^*, C_3^* and C_4^* are constructed as follows: $\mathcal{C}_2$ chooses $u, v \in_R \mathbb{Z}_p$ such that $u \neq v$, sets $U = g_3^u$ and $V = g_4^v$, and computes $C_1^* = A_{i_b} U^{z_1} V^{z_2}$, $C_2^* = U$, $C_3^* = V$ and $C_4^* = U^{x_1' + y_1' \beta^*} V^{x_2' + y_2' \beta^*}$, where $\beta^* = \mathcal{H}(C_1^*, C_2^*, C_3^*)$. If $u = v$, then $C_1^* = A_{i_b} g_3^{u z_1} \cdot g_4^{v z_2} = A_{i_b} E^u$, $C_2^* = g_3^u$, $C_3^* = g_4^v = g_4^u$, and $C_4^* = (g_3^{x_1'} g_4^{x_2'})^u (g_3^{y_1'} g_4^{y_2'})^{v\beta^*} = (CD^{\beta^*})^u$ hold. This is the same as Game 1. Therefore, obviously $|\Pr[S_1] - \Pr[S_2]| = \epsilon_{\mathrm{ddh}}$ holds.

From Game 3 to Game $\phi + 1$. Game j ($j = 3, \ldots, \phi + 1$) is the same as Game $j-1$ except CT_{j-2}^* and CT_{j-1}^* are constructed as follows: $\mathcal{C}_j$ chooses $u_j, v_j \in_R \mathbb{Z}_p$ such that $u_j \neq v_j$. $\mathcal{C}_j$ computes $CT_{j-2}^* = T_{i_b, j-2} \hat{h}_{j-2}^{u_j}$ and $CT_{j-1}^* = T_{i_b, j-1} \hat{h}_{j-2}^{v_j}$. If $u_j = v_j$, then $CT_{j-2}^* = T_{i_b, j-2} \hat{h}_{j-2}^{u_j}$ and $CT_{j-1}^* = T_{i_b, j-1} \hat{h}_{j-1}^{u_j}$ hold. This is the same as Game $j-1$. Therefore, obviously $|\Pr[S_{j-1}] - \Pr[S_j]| = \epsilon_{\mathrm{ddh}}$ holds.

Note that $\Pr[S_{\phi+1}] = \frac{1}{2}$ because all parts of the challenge group signature in the case of U_{i_0} and all parts of the challenge group signature in the case of U_{i_1} have the same distributions. Combining all the probabilistic relations from Game 0 to Game $\phi + 1$, $Adv^{anon}(\mathcal{A}) = \Pr[S_0] - \frac{1}{2} \leq \frac{q_S q_H}{p} + \phi \cdot \epsilon_{\mathrm{ddh}} \leq \frac{q_S q_H}{p} + m \cdot \epsilon_{\mathrm{ddh}}$ holds, where $m = |Att|$.

Next, we give the proof of Theorem 2 as follows:

Proof 2 We assume that the challenge attributes ζ^* satisfy the challenge access tree T^*, namely, $\zeta^* \models T^*$. If $\zeta^* \not\models T^*$, then the probability of the signature made by forged attribute certificates accepting the verification is negligible (See Theorem 4). The input of simulator $\mathcal{B}$ is $(g, g', (g')^{\xi}, \ldots, (g')^{\xi^q}) \in \mathbb{G}_1 \times \mathbb{G}_2^{q+1}$. Let $q-1$ be the number of all members, n be the number of honest members, and $q_1 = q-1-n$ be the number of corrupted members. We assume that all initial members $\{U_1, \ldots, U_n\}$ are honest. $\mathcal{B}$ simulates KeyGen as follows.

(1) $\mathcal{B}$ selects $\mu \in_R \mathbb{Z}_p^*$, $x_i \in_R \mathbb{Z}_p^*$ $(i = 1, 2, \ldots, q-1)$, $y_i \in_R \mathbb{Z}_p^*$ $(i = 1, 2, \ldots, n)$, x_1', x_2', y_1', y_2', $z \in \mathbb{Z}_p$, $g_4 \in \mathbb{G}_1$ and $h_j \in \mathbb{G}_2$ $(att_j \in Att)$.

(2) $\mathcal{B}$ selects a target user $U_{i^*} \in \{U_1, \ldots, U_{q-1}\}$, and sets $\gamma := \xi - x_{i^*}$. $\mathcal{B}$ computes g_1, g_2, g_3 and w as follows:

$$\begin{aligned} g_2 &:= (g')^{\mu \prod_{i=1}^{q-1}(\xi + x_i - x_{i^*})} / (g')^{z y_{i^*} \prod_{i=1, i \neq i^*}^{q-1}(\xi + x_i - x_{i^*})} \\ g_1 &:= \psi(g_2) = (g_3^{\mu\xi} / E^{y_i^*}) \\ g_3 &:= g^{\prod_{i=1, i \neq i^*}^{q-1}(\xi + x_i - x_{i^*})} \\ w &:= \left\{ (g')^{\mu\xi \prod_{i=1}^{q-1}(\xi + x_i - x_{i^*})} \Big/ (g')^{z y_{i^*} \xi \prod_{i=1, i \neq i^*}^{q-1}(\xi + x_i - x_{i^*})} \right\} \Big/ g_2^{x_{i^*}} \\ &= g_2^{\xi - x_{i^*}} \\ &= g_2^{\gamma} \end{aligned}$$

$\mathcal{B}$ can compute these values by using the q-SDH input instance, because all parts of the exponent can be expressed as a polynomial of degree at most q.

(3) $\mathcal{B}$ computes $C = g_3^{x_1'} g_4^{x_2'}$, $D = g_3^{y_1'} g_4^{y_2'}$, $E = g_3^{z}$ and other parameters.

(4) $\mathcal{B}$ makes $params = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_3, e, \psi, \mathcal{H}, g_1, g_2, g_3, g_4, Att)$, $ok = (z)$, $ik = (\gamma, \{s_j\}_{att_j \in Att})$, $gpk = (\omega, C, D, E, \{h_j\}_{j=1}^{m}, \{g_{att_j}\}_{att_j \in Att})$ and $\mathcal{T}_0 = (\{g_{d_j}\}_{d_j \in D_{\mathcal{T}_0}}, v_0, T_0^{ext})$. $params$, $\mathcal{T}_0$, gpk and ok are given to $\mathcal{A}$.

In the Join queries, $\mathcal{B}$ can get a secret value y of a corrupted user to extract the commitment value. $\mathcal{B}$ computes a group membership certificate as follows:

In the case of $i = i^*$: $\mathcal{B}$ computes $A_{i^*} = g_3^{\mu} = (g_3^{\mu\xi})^{\frac{1}{\xi}} = (g_1 E^{y_{i^*}})^{\frac{1}{\gamma + x_{i^*}}}$.

In the case of $i \neq i^*$: $\mathcal{B}$ computes A_i as follows:

$$\begin{aligned} A_i &= \left(g^{z \prod_{j=1, j \neq i^*, i}^{q-1}(\xi + x_j - x_{i^*})} \right)^{y_i - y_{i^*}} \cdot g^{\mu \prod_{j=1, j \neq i}^{q-1}(\xi + x_j - x_{i^*})} \\ &= g^{\frac{z y_i}{\xi + x_i - x_{i^*}} \prod_{j=1, j \neq i^*}^{q-1}(\xi + x_j - x_{i^*})} \\ &\quad \times \left\{ g^{\mu \prod_{j=1}^{q-1}(\xi + x_j - x_{i^*})} \Big/ g^{z y_{i^*} \prod_{j=1, j \neq i^*}^{q-1}(\xi + x_j - x_{i^*})} \right\}^{\frac{1}{\xi + x_i - x_{i^*}}} \\ &= (g_1 E^{y_i})^{\frac{1}{\gamma + x_i}} \end{aligned}$$

$\mathcal{B}$ can choose $s_j \in \mathbb{Z}_p$ $(att_j \in Att)$. Therefore $\mathcal{B}$ can compute $\{T_{i,j}\}_{att_j \in \Gamma_i} = \{A_i^{s_j}\}_{att_j \in \Gamma_i}$. For signing queries, $\mathcal{B}$ makes a group signature by using $(A_i, x_i, y_i, \{T_{i,j}\}_{att_j \in \Gamma_i})$, and returns this signature. For corruption queries, $\mathcal{B}$ answers $(A_i, x_i, y_i, \{T_{i,j}\}_{att_j \in \Gamma_i})$. Re-BuildTree queries are the same as the proof of anonymity. Finally, $\mathcal{A}$ outputs a forged signature $\sigma^* = (C_1^*, C_2^*, C_3^*, C_4^*, \{CT_j^*\}_{att_j \in \zeta^*}, c^*, s_\alpha^*, s_\delta^*, s_x^*, s_\tau^*)$.

By using the Forking Lemma, $\mathcal{B}$ can get the two valid signatures $(C_1^*, C_2^*, C_3^*, C_4^*, \{CT_j^*\}_{att_j \in \zeta^*}, c^*, s_\alpha^*, s_\delta^*, s_x^*, s_\tau^*)$ and $(C_1^*, C_2^*, C_3^*, C_4^*, \{CT_j^*\}_{att_j \in \zeta^*}, c', s_\alpha', s_\delta', s_x', s_\tau')$ with probability $\epsilon' \geq \frac{1}{5} - \frac{8 q_H}{\eta 2^k}$, $\eta > \frac{240 q_H}{2^k}$ 14). Let $c'' = c^* - c'$, $s_\alpha'' = s_\alpha^* - s_\alpha'$, $s_x'' = s_x^* - s_x'$, and $s_\tau'' = s_\tau^* - s_\tau'$. Let $\tilde{x} = s_x''/c''$, $\tilde{\alpha} = s_\alpha''/c''$, $\tilde{\tau} = s_\tau''/c''$, $\tilde{A} = C_1^*/E^{\tilde{\alpha}}$, and $\tilde{y} = \tilde{\tau} - \tilde{\alpha}\tilde{x}$.

Now $(\tilde{A}, \tilde{x}, \tilde{y})$ is a valid member certificate because $e(\tilde{A}, g_2)^{\tilde{x}} e(\tilde{A}, w) e(E, g_2)^{-\tilde{y}} = e(g_1, g_2)$ holds. We assume that $\tilde{x} \neq x_{i^*}$. This probability is $1 - \frac{1}{p}$.

$$\begin{aligned} \tilde{A} &= (g_1 E^{\tilde{y}})^{\frac{1}{\tilde{x} + \gamma}} \\ &= (g_3^{\mu\xi} E^{\tilde{y} - y_{i^*}})^{\frac{1}{\tilde{x} + \gamma}} \\ &= g_3^{\frac{\mu\xi + z(\tilde{y} - y_{i^*})}{\tilde{x} + \gamma}} \\ &= \left(g^{(\mu\xi + z(\tilde{y} - y_{i^*})) \prod_{i=1, i \neq i^*}^{q-1}(\xi + x_i - x_{i^*})} \right)^{\frac{1}{\tilde{x} + \xi - x_{i^*}}} \\ &= \left(g^{\sum_{i=0}^{q-1} a_i \xi^i} \right)^{\frac{1}{\tilde{x} + \xi - x_{i^*}}} \\ &= g^{\frac{b_0}{\tilde{x} + \xi - x_{i^*}} + \sum_{i=1}^{q-1} b_i \xi^i} \end{aligned}$$

The polynomial coefficients $a_0, \ldots, a_{q-1}, b_0, b_1, \ldots, b_{q-1}$ can be computed by $\mathcal{B}$. Let $x = \tilde{x} - x_{i^*}$. Then $(\tilde{A}/g^{\sum_{i=1}^{q-1} b_i \xi^i})^{\frac{1}{b_0}} = g^{\frac{1}{x+\xi}}$ holds. Therefore $(x, g^{\frac{1}{x+\xi}})$ is the new SDH tuple. The advantage of $\mathcal{B}$ is $(1 - \frac{1}{p})(1 - \frac{q_S q_H}{p})(\frac{1}{5} - \frac{8 q_H}{\eta 2^k})\epsilon \geq$

$\frac{1}{6}(1-\frac{1}{p})(1-\frac{q_S q_H}{p})\epsilon$, since $\eta > \frac{240q_H}{2^k}$.

Next, we give the proof of Theorem 3 as follows:

Proof 3 The input of simulator $\mathcal{B}$ is $(g, g') \in \mathbb{G}_2 \times \mathbb{G}_2$. We consider the two types of adversaries by the results of the Open algorithm. We explain the details of classification of the adversary in the proof. Let q be the number of all members, n be the number of honest members, and $q_1 = q - n$ be the number of corrupt members. We assume that all initial members $\{U_1, \ldots, U_n\}$ are honest. $\mathcal{B}$ simulates KeyGen as follows.

(1) $\mathcal{B}$ selects $d \in_R \{0, 1\}$, $z \in \mathbb{Z}_p$ and $s_j \in \mathbb{Z}_p$ ($att_j \in Att$). If $d = 1$, then $\mathcal{B}$ selects a target user $U_{i^*} \in \{U_1, \ldots, U_n\}$. $d = 0$ means $\mathcal{B}$ guesses that $\mathcal{A}$ is a Type 1 Adversary. And $d = 1$ means $\mathcal{B}$ guesses that $\mathcal{A}$ is a Type 2 Adversary.

(2) $\mathcal{B}$ computes the group public key and member certificates as follows:
  - (a) $\mathcal{B}$ selects $\gamma \in_R \mathbb{Z}_p^*$ and $x_i, y_i \in \mathbb{Z}_p^*$ ($i \in [1, q]$). If $d = 1$, then $i = i^*$ is excepted from the above $[1, q]$.
  - (b) If $d = 0$, then $\mathcal{B}$ sets $g_1 = \psi(g)$, $g_2 = g$ and $g_3 = \psi(g')$, and compute $w = g_2^{\gamma}$ and $E = g_3^{z}$.
  - (c) If $d = 1$, then $\mathcal{B}$ sets $g_2 \in_R \mathbb{G}_2$, $g_1 = \psi(g_1)$, $g_3 = g$, and $y_{i^*} = \xi$.
  - (d) $\mathcal{B}$ computes (A_i, x_i, y_i) ($i \in [1, q]$) by using γ. If $d = 1$, then $i = i^*$ is excepted from the above $[1, q]$.
  - (e) $\mathcal{B}$ computes other public values, and gets *params* and *gpk*.

(3) $\mathcal{B}$ gives *params*, *gpk*, $ik = (\gamma, \{s_j\}_{att_j \in Att})$ and $ok = (z)$ to $\mathcal{A}$.

In Join queries, $\mathcal{A}$ knows (A_i, x_i) ($i = 1, \ldots, q$) because $\mathcal{A}$ plays the role of corrupted GM. However, $\mathcal{A}$ cannot know secret keys of honest users y_i ($i = 1, \ldots, n$). For Signing queries, $\mathcal{B}$ makes a group signature by using (A_i, x_i, y_i), and returns this signature. If $d = 1$ and $i = i^*$, then $\mathcal{B}$ aborts. For Corruption queries, $\mathcal{B}$ answers y_i. If $d = 1$ and $i = i^*$, then $\mathcal{B}$ aborts. Re-BuildTree queries are the same as the proof of anonymity. Finally, $\mathcal{A}$ outputs the valid group signature for honest user U_k. We can get the member certificate (A^*, x^*, y^*) by using the same technique as for Traceability. We define a Type 1 Adversary $\mathcal{A}$, which is the case of $A^* = A_k \in \{A_i\}_{i=1}^n$ and $x^* \neq x_k$. We define a Type 2 Adversary $\mathcal{A}$, which is the case of $(A^*, x^*) = (A_k, x_k)$.

- In the case of Type 1 : If $d \neq 0$, then $\mathcal{B}$ aborts. Otherwise $A^* = (g_1 E^{y^*})^{\frac{1}{x^*+\gamma}} = (g_1^{1+z\xi y^*})^{\frac{1}{x^*+\gamma}}$ holds. $A^* = A_k = (g_1 E^{y_k})^{\frac{1}{x_k+\gamma}} = (g_1^{1+z\xi y_k})^{\frac{1}{x_k+\gamma}}$ holds. Therefore $\mathcal{B}$ can compute $\xi = \frac{x^*-x_k}{z\{y^*(x_k+\gamma)-y_k(x^*+\gamma)\}}$.
- In the case of Type 2 : If $d \neq 1$, then $\mathcal{B}$ aborts. If $k \neq i^*$, then $\mathcal{B}$ aborts. Otherwise, $A^* = (g_1 E^{y^*})^{\frac{1}{x^*+\gamma}} = (g_1 g^{zy^*})^{\frac{1}{x^*+\gamma}}$ holds. Moreover, $A^* = A_{i^*} = (g_1 E^{y_{i^*}})^{\frac{1}{x_{i^*}+\gamma}} = (g_1 g_3^{z\xi})^{\frac{1}{x_{i^*}+\gamma}}$ holds. Therefore $\mathcal{B}$ can get $\xi = y^*$.

The advantage of $\mathcal{B}$ is $(1-\frac{q_S q_H}{p})(\frac{1}{2}(\frac{1}{5}-\frac{8q_H}{\eta 2^k})\epsilon + \frac{1}{2}\frac{1}{n}(\frac{1}{5}-\frac{8q_H}{\eta 2^k})\epsilon) \geq \frac{1}{12}(1+\frac{1}{n})(1-\frac{q_S q_H}{p})\epsilon$, since $\eta > \frac{240q_H}{2^k}$.

Next, we give the proof of Theorem 4 as follows:

Proof 4 We assume that $\zeta_i = \{att_1, \ldots, att_\phi\}$, without limiting the generality of the foregoing. The equations used in our scheme to prove the knowledge of $(\alpha, x_i, \tau, \delta)$ are as follows:

$$\frac{e(C_1, \omega)}{e(g_1, g_2)} = \frac{e(E, g_2)^{\tau} \cdot e(E, \omega)^{\alpha}}{e(C_1, g_2)^{x_i}} \quad (1)$$

$$C_2 = g_3^{\alpha} \quad (2)$$

$$C_3 = g_4^{\alpha} \quad (3)$$

$$C_4 = (CD^{\beta})^{\alpha} \quad (4)$$

$$\frac{e(\prod_{att_j \in \zeta_i} CT_j^{\Delta_j}, g_2)}{e(C_1, v_r/g_d)} = \frac{e(\prod_{att_j \in \zeta_i} \hat{h}_j^{\Delta_j}, g_2)^{\delta}}{e(E, v_r/g_d)^{\alpha}} \quad (5)$$

In Eq. (1), a signer proves that $C_1 = E^{\alpha} A_i$, where A_i is a valid member certificate[14),24)]. Equations (2), (3) and (4) obviously holds. We can change Eq. (5) into $\frac{e(\prod_{att_j \in \zeta_i} CT_j^{\Delta_j}, g_2)}{e(E^{\alpha} A_i, v_r/g_d)} = \frac{e(\prod_{att_j \in \zeta_i} \hat{h}_j^{\Delta_j}, g_2)^{\delta}}{e(E, v_r/g_d)^{\alpha}}$, since the validity of SPK C_1 (namely (1)) has already been proven. $e(\prod_{att_j \in \zeta_i} (\hat{h}_j^{-\delta} CT_j)^{\Delta_j}, g_2) = e(A_i, v_r) e(A_i, g_d)^{-1} = e(A_i^{s_{T_r} - \sum_{d_j \in D_r^{\zeta}} \Delta_{d_j} s_{d_j}}, g_2)$ holds. We assume that $\hat{h}_j^{-\delta} CT_j = A_i^{t_j}$, where $t_j \in \mathbb{Z}_p$. Then

$$s_{T_r} = \sum_{att_j \in \zeta_i} \Delta_j t_j + \sum_{d_j \in D_r^{\zeta_i}} \Delta_{d_j} s_{d_j} \quad (6)$$

holds, since $\prod_{att_j \in \zeta_i} (\hat{h}_j^{-\delta} CT_j)^{\Delta_j} = A_i^{\sum_{att_j \in \zeta_i} \Delta_j t_j}$. If $t_j = s_j$ ($att_j \in \zeta_i$), then (6) obviously holds. On the contrary, we assume that t_j ($att_j \in \zeta_i$) satisfies

Table 1 Comparisons.

	Reference 18)	Reference 17)	Our Scheme
Dynamic property	no	no	yes
CCA-Anonymity	no	no	yes
Non-Frameability	no	no	yes
Key-Exposure	no	no	yes
Signature Length	$1633 + 171\phi$	$1192 + 1191\phi$	$1634 + 171\phi$
PK Length	$(m+3)\lvert\mathbb{G}_1\rvert + (m+1)\lvert\mathbb{G}_2\rvert$	$2\lvert\mathbb{G}_1\rvert+$	$(m+3)\lvert\mathbb{G}_1\rvert + (2m+1)\lvert\mathbb{G}_2\rvert(m+1)\lvert\mathbb{G}_2\rvert$
User's SK Length	$\lvert\mathbb{Z}_p\rvert + (m'+1)\lvert\mathbb{G}_1\rvert$	$\lvert\mathbb{Z}_p\rvert + (m'+1)\lvert\mathbb{G}_1\rvert$	$2\lvert\mathbb{Z}_p\rvert + (m'+1)\lvert\mathbb{G}_1\rvert$
Signing	$(12+2\phi)\mathbb{G}_1 + 5\mathbb{G}_3 + e$	$(7+2\phi)\mathbb{G}_1 + (5+\phi)\mathbb{G}_3 + (\phi+1)e$	$(9+3\phi)\mathbb{G}_1 + (\phi+1)\mathbb{G}_2 + 8\mathbb{G}_3 + 3e$
Verification	$12\mathbb{G}_1 + (\phi+8)\mathbb{G}_3 + (\phi+1)e$	$(6+2r)\mathbb{G}_1 + (8+2\phi)\mathbb{G}_3 + (\phi+2r+1)e$	$(11+2\phi)\mathbb{G}_1 + (\phi+1)\mathbb{G}_2 + 14\mathbb{G}_3 + 6e$

(6). We set the values of $s_{T_r}, \Delta_j, \Delta_{d_j}$ and s_{d_j} as constants. We randomly choose $t_j \in \mathbb{Z}_p$ ($j = 1, 2, \ldots, \phi-1$), and set $t_\phi := (s_{T_r} - \sum_{att_j \in \zeta_i \setminus \{att_\phi\}} \Delta_j t_j - \sum_{d_j \in D_r^{\zeta_i}} \Delta_{d_j} s_{d_j})/\Delta_\phi$. Then $(t_1, \ldots, t_\phi)$ satisfies (6). Therefore, the number of the solution vectors $(t_1, t_2, \ldots, t_\phi)$ is $p^{\phi-1}$. Therefore, the probability of the randomly chosen vector $(t_1, t_2, \ldots, t_\phi)$ satisfying (6) is $p^{\phi-1}/p^\phi = 1/p$. This implies that, the probability of a signature made by forged attribute certificates satisfying (6) is $\frac{p^{\phi-1}-1}{p^\phi} = \frac{1}{p}(1 - \frac{1}{p^{\phi-1}})$. Next, we consider $t_j = s_j$ ($j = 1, 2, \ldots, \ell$), where $\ell < \phi$. Let $\ell = \phi - 1$, this means a signer has valid attribute certificates of $\zeta_i \setminus \{att_\phi\}$. We assume that a signature satisfies (5). Then $t_\phi := (s_{T_r} - \sum_{att_j \in \zeta_i \setminus \{att_\phi\}} \Delta_j s_j - \sum_{d_j \in D_r^{\zeta_i}} \Delta_{d_j} s_{d_j})/\Delta_\phi = s_\phi$ hold. This means that the signer has valid attribute certificates of ζ_i, and the signature is not a forged signature. Therefore, we set $\ell < \phi - 1$. This means a signer has valid attribute certificates of $\zeta_i \setminus \{att_{\ell+1}, \ldots, att_\phi\}$. Then there exist the number of $p^{\phi-\ell-1} - 1$ pair $(t_{\ell+1}, \ldots, t_\phi)$ such that $(s_1, \ldots, s_\ell, t_{\ell+1}, \ldots, t_\phi)$ satisfies (6) and $(t_{\ell+1}, \ldots, t_\phi) \neq (s_{\ell+1}, \ldots, s_\phi)$. The number of vectors $(t_{\ell+1}, \ldots, t_\phi)$ is $p^{\phi-\ell}$. Therefore, the probability of a signature made by valid attribute certificates of $\zeta_i \setminus \{att_{\ell+1}, \ldots, att_\phi\}$ and forged attribute certificates of $\{att_{\ell+1}, \ldots, att_\phi\}$ satisfying (6) is $\frac{p^{\phi-\ell-1}-1}{p^{\phi-\ell}} = \frac{1}{p}(1 - \frac{1}{p^{\phi-\ell-1}}) \leq \frac{1}{p}$.

So, a verifier can decide whether an anonymous signer has valid attribute certificates when a verifier is given a signature which satisfies (5).

Next, we give the proof of Theorem 5. The equations used in the proof are the same as in the proof of Theorem 4.

Proof 5 Without loss of generality, we assume that U_0 with ζ_0 and U_1 with ζ_1 represent malicious participants. U_0 and U_1 attempt to make a valid signature associated with T_r which satisfies $\zeta_0 \cup \zeta_1 \models T_r$, $\zeta_0 \not\models T_r$ and $\zeta_1 \not\models T_r$. They can make the SPK of $(\alpha, x_0, \tau, \delta)$ satisfy Eqs. (1) to (4) because they have a valid membership certificate A_0. Let $\zeta_0 \cup \zeta_1 := \zeta \models T_r$. We assume that $A_0^t = A_1$, where $t \in \mathbb{Z}_p^* \setminus \{1\}$. Note that the probability of $t = 1$ is negligible. Then, from (6), $\sum_{att_j \in \zeta_0} \Delta_j s_j + \sum_{att_j \in \zeta_1} t\Delta_j s_j + \sum_{d_j \in D_r^\zeta} \Delta_{d_j} s_{d_j} \neq s_{T_r}$ holds since $t \neq 1$. This means that they cannot use $\{T_{i_0,j}\}_{att_j \in \zeta_0}$ and $\{T_{i_1,j}\}_{att_j \in \zeta_1}$ simultaneously. Even if they attempt to make forged attribute certificates, the probability of accepting a signature is negligible from Theorem 4.

5. Comparisons

In this section, we compare the efficiency of our proposed scheme with previous ABGS schemes[17),18)]. To the best of our knowledge, these are the only proposals for ABGS. Let ζ ($|\zeta| = \phi$) be the set of attributes which is associated with a signature, D_ζ be the set of dummy nodes which is defined as ζ, and $|Att| = m$. We evaluate $|D_\zeta| = O(|Att|)$ and treat $|D_\zeta| \approx |\zeta| = \phi$ in **Table 1**. Let $m' \leq m$ be the number of attributes for each user. Actually, m' is different for each user. However, to simplify, we use the same notation m' to express the bit-length of a user's secret key. Let r be the number of revoked members[17)]. We assume that the computational estimations are made according to NF06 scheme[24)], i.e., using MNT curves[23)]. The prime number p is 170 bits, elements of $\mathbb{G}_1$ are 171 bits, elements of $\mathbb{G}_2$ are 513 bits, and elements of $\mathbb{G}_3$ are 1020 bits. In our scheme, although Signing costs are higher than that of a previous scheme[18], Verification costs are the lowest, because the number of calculations in a pairing does not depend on the number of attributes associated with a signature. There is room for argument regarding the Signing costs. Our scheme provides dynamic property, CCA-anonymity, key-exposure and Non-Frameability, which were not provided in the previous ABGSs.

6. Application of ABGS in Anonymous Survey for Collection of Attribute Statistics

In this section, we discuss how our ABGS can be applied to an anonymous survey for the collection of attribute statistics. An anonymous survey is used as follows: When we apply the GS to a business system offering some services to group members, each member's personal information is not exposed. A service provider can verify whether each user is valid or not. However, it is difficult for a service provider to obtain a collection of the user's attribute statistics to improve service contents. To apply an anonymous survey, a service provider can obtain a collection of user's attribute statistics without exposing each user's information. Although an anonymous attribute authentication scheme has been proposed[29], this scheme treats only one attribute on a single authentication execution. This means the relationships among some attributes, e.g., (female $\wedge$ 20 s), cannot be handled in the statistical information. An anonymous survey which is a protocol executed among trusted third parties (TTPs) has been proposed[30]. Each TTP is associated with one attribute. A user with some attributes sends the distributor a ciphertext encrypted with the public key of the TTP who is in charge of user's attribute type. The distributor can obtain the statistics of attributes without any other information to execute this protocol. The relationships among some attributes can be handled in the statistical information. However, a distributor cannot verify whether users properly construct the ciphertext or not. An anonymous survey which is a protocol using the Open algorithm of Ateniese, et al. GS[2] has been proposed[25]. A distributor can verify whether users properly make the ciphertext or not, to verify the validity of group signatures. In the NS03 scheme[25], each user has attribute certificates which are used for making a group signature. The distributor executes the Open algorithm to reveal the signer's attribute type. Because one attribute certificate is issued for an attribute type, it is difficult for the relationships among some attributes to be handled in the statistical information. There is an obvious solution: new attribute types such as $att_C = att_A \wedge att_B$ are defined. However, the number of all attribute types are represented by $O(2^m)$, where m is the number of all attributes. We solve this attribute increase problem to apply an ABGS.

(1) A user makes a group signature σ associated with the set of attributes ζ to use our ABGS.
(2) The user encrypts ζ to use the public key of a distributor, and sends both σ and the encrypted ζ to the distributor.
(3) The distributor decrypts ζ, and verifies whether σ is valid or not.
(4) The statistical information is the collection of ζ.

To collect the set of attributes ζ, the distributor can obtain the statistics of attributes without any other information, because the distributor does not know the opening key which is used for the opening procedure to reveal the signer's identification from the group signature. The distributor can verify whether users properly made the ciphertext or not, to verify that the validity of group signatures is the same as in the NS03 scheme[25]. The relationships among some attributes can be handled in the statistical information in the same way as in the SAKO96 scheme[30], without increasing the number of attribute certificates of each user. Indeed, the number of attribute certificates of each user is represented by $O(m)$. Of course, relationships among some attributes which one wants to reflect with the statistical information are different in each case. Our scheme is suitable for use in the anonymous survey because the change of relationships is indispensable in the anonymous survey for the collection of attribute statistics.

7. Conclusion

In this paper, we propose a dynamic ABGS scheme that enables an access tree to be changed. Our ABGS is efficient in that re-issuing of the attribute certificate previously issued for each user is not necessary. As minor contributions, our ABGS enables CCA-anonymity and key-exposure properties, and the number of calculations in a pairing does not depend on the number of attributes associated with a signature. A service provider obtains a collection of anonymous user's attribute statistics to improve service contents by using our ABGS.

References

1) Ateniese, G., Blanton, M. and Kirsch, J.: Secret Handshakes with Dynamic and Fuzzy Matching, *Network & Distributed System Security Symposium, NDSS 2007*, pp.159–177 (2007).
2) Ateniese, G., Camenisch, J., Joye, M. and Tsudik, G.: A practical and provably secure coalition-resistant group signature scheme, *CRYPTO 2000*, pp.255–270 (2000).
3) Ateniese, G. and Tsudik, G.: Some open issues and new directions in group signatures, *FC '99: Proc. 3rd International Conference on Financial Cryptography*, pp.196–211, Springer-Verlag, London, UK (1999).
4) Boneh, D. and Boyen, X.: Short signatures without random oracles, *EUROCRYPT 2004*, pp.56–73 (2004).
5) Boneh, D., Boyen, X. and Shacham, H.: Short group signatures, *CRYPTO 2004*, pp.41–55 (2004).
6) Boneh, D. and Franklin, M.: Identity-based encryption from the weil pairing, pp.213–229, Springer-Verlag (2001).
7) Bellare, M., Shi, H. and Zhang, C.: Foundations of group signatures: The case of dynamic groups, *CT-RSA 2005*, pp.136–153 (2005).
8) Bender, A., Katz, J. and Morselli, R.: Ring signatures: Stronger definitions, and constructions without random oracles, *Theory of Cryptography Conference, TCC 2006*, pp.60–79 (2006)
9) Bethencourt, J., Sahai, A. and Waters, B.: Ciphertext-policy attribute-based encryption, *IEEE Symposium on Security and Privacy*, pp.321–334 (2007).
10) Boyen, X. and Waters, B.: Anonymous hierarchical identity-based encryption (without random oracles), *CRYPTO*, pp.290–307, Springer-Verlag (2006).
11) Chase, M.: Multi-authority attribute based encryption, *TCC*, pp.515–534 (2007).
12) Cheung, L. and Newport, C.: Provably secure ciphertext policy ABE, *CCS '07: Proc. 14th ACM Conference on Computer and Communications Security*, pp.456–465, ACM (2007).
13) Cramer, R. and Shoup, V.: A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack, *CRYPTO '98*, pp.13–25 (1998).
14) Delerablée, C. and Pointcheval, D.: Dynamic fully anonymous short group signatures, *VIETCRYPT 2006*, pp.193–210 (2006).
15) Goyal, V., Jain, A., Pandey, O. and Sahai, A.: Bounded ciphertext policy attribute based encryption, *The 35th International Colloquium on Automata, Languages and Programming, ICALP*, pp.579–591 (2008).
16) Goyal, V., Pandey, O., Sahai, A. and Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data, *ACM CCS 2006*, pp.89–98 (2006).
17) Khader, D.: Attribute based group signature with revocation, Cryptology ePrint Archive, Report 2007/241 (2007).
18) Khader, D.: Attribute based group signatures, Cryptology ePrint Archive, Report 2007/159 (2007).
19) Lin, H., Cao, Z., Liang, X. and Shao, J.: Secure threshold multi authority attribute based encryption without a central authority, *INDOCRYPT*, pp.426–436 (2008).
20) Lubicz, D. and Sirvent, T.: Attribute-based broadcast encryption scheme made efficient, *AFRICACRYPT*, pp.325–342 (2008).
21) Mao, W.: *Modern Cryptography: Theory and Practice*, Prentice Hall Professional Technical Reference (2003).
22) Müller, S., Katzenbeisser, S. and Eckert, C.: Distributed attribute-based encryption, *11th International Conference on Information Security and Cryptology, ICISC '08*, pp.20–36 (2008).
23) Miyaji, A., Nakabayashi, M. and Takano, S.: New explicit conditions of elliptic curve traces for fr-reduction, *IEICE Trans. Fundamentals of Electronics, Communications and Computer Sciences*, Vol.84, No.5, pp.1234–1243 (2001).
24) Nakanishi, T. and Funabiki, N.: A short verifier-local revocation group signature scheme with backward unlinkability, *International Workshop on Security, IWSEC 2006*, pp.17–32 (2006).
25) Nakanishi, T. and Sugiyama, Y.: An efficient anonymous survey for attribute statistics using a group signature scheme with attribute tracing, *IEICE Trans. Fundamentals of Electronics, Communications and Computer Sciences*, Vol.86, No.10, pp.2560–2568 (2003).
26) Nishide, T., Yoneyama, K. and Ohta, K.: Attribute-based encryption with partially hidden encryptor-specified access structures, *ACNS*, pp.111–129 (2008).
27) Nakanishi, T., Tao, M. and Sugiyama, Y.: A group signature scheme committing the group, *ICICS '02*, pp.73–84 (2002).
28) Paillier, P.: Public-key cryptosystems based on discrete logarithms residues, *EUROCRYPT '99*, pp.223–238 (1999).
29) Kiyomoto, S. and Tanaka, T.: Anonymous Attribute Authentication Scheme Using Self-Blindable Certificates, *IEEE Intelligence and Security Informatics, ISI 2008*, pp.215–217 (2008).
30) Sako, K.: Generating statistical information in anonymous surveys, *IEICE Trans. Fundamentals of Electronics, Communications and Computer Sciences*, Vol.79, No.4, pp.507–512 (1996).
31) Shoup, V.: Sequences of games: A tool for taming complexity in security proofs, Cryptology ePrint Report 2004/332 (Nov. 2004).
32) Sahai, A. and Waters, B.: Fuzzy identity-based encryption, *EUROCRYPT*, pp.457–473 (2005).
33) Trolin, M. and Wikström, D.: Hierarchical group signatures, *ICALP '05*, pp.446–458 (2005).
34) Waters, B.: Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization, Cryptology ePrint Report 2008/290 (Sep. 2008).

(Received December 1, 2008)
(Accepted June 4, 2009)
(Released September 9, 2009)

Keita Emura received the B.E. and M.E. degrees from Kanazawa University, Ishikawa, Japan in 2002 and 2004, respectively. He worked at Fujitsu Hokuriku Systems Limited from 2004 to 2006, and was engaged in research and development for asynchronous communication. He is currently pursuing a doctorate degree at Japan Advanced Institute of Science and Technology (JAIST). His research interests include cryptography and information security.

Atsuko Miyaji received the B. Sc., the M. Sc., and the Dr. Sci. degrees in mathematics from Osaka University, Osaka, Japan in 1988, 1990, and 1997 respectively. She joined Panasonic Co., LTD from 1990 to 1998 and engaged in research and development for secure communication. She was an associate professor at the Japan Advanced Institute of Science and Technology (JAIST) in 1998. She has joined the computer science department of the University of California, Davis since 2002. She has been a professor at the Japan Advanced Institute of Science and Technology (JAIST) since 2007 and the director of Library of JAIST since 2008. Her research interests include the application of number theory into cryptography and information security. She received the IPSJ Sakai Special Researcher Award in 2002, the Standardization Contribution Award in 2003, Engineering Sciences Society: Certificate of Appreciation in 2005, the AWARD for the contribution to CULTURE of SECURITY in 2007, IPSJ/ITSCJ Project Editor Award in 2007, 2008, and 2009, the Director-General of Industrial Science and Technology Policy and Environment Bureau Award in 2007, Editorial Committee of Engineering Sciences Society: Certificate of Appreciation in 2007, and DoCoMo Mobile Science Awards in 2008. She is a member of the International Association for Cryptologic Research, the Institute of Electronics, Information and Communication Engineers, the Information Processing Society of Japan, and the Mathematical Society of Japan.

Kazumasa Omote received his M.S. and Ph.D. degrees in information science from Japan Advanced Institute of Science and Technology (JAIST) in 1999 and 2002, respectively. He worked at Fujitsu Laboratories, LTD from 2002 to 2008, and was engaged in research and development for network security. He has been a research assistant professor at Japan Advanced Institute of Science and Technology (JAIST) since 2008. His research interests include applied cryptography and network security. He is a member of the IPS of Japan.